uc3m | Universidad Carlos III de Madrid

Escuela Politécnica Superior

Grado en Ingeniería en Tecnologías de Telecomunicación

Trabajo Fin de Grado

"Evaluación del protocolo Bundle mediante un emulador de redes"

Autor: Carlos Díez Rodríguez

Tutor: Ignacio Soto Campos

Leganés, junio 2019

RESUMEN

Las redes de comunicaciones han adquirido una gran importancia a lo largo del tiempo debido a la necesidad del ser humano de descubrir y tener un mayor conocimiento sobre todo lo que nos rodea. Actualmente, las zonas urbanas y las zonas rurales de países desarrollados se encuentran bien comunicadas debido a la red tradicional de cable coaxial o fibra óptica que se encuentra funcionando gracias a la torre de protocolos TCP/IP. Sin embargo, siguen existiendo zonas de nuestro planeta situadas por ejemplo en países subdesarrollados o en entornos extremos como grandes montañas o zonas cercanas a volcanes que también tienen la necesidad de comunicarse. Además, las redes tradicionales no tienen un buen funcionamiento cuando se trata de comunicaciones desde la tierra con el espacio exterior debido a las largas distancias y al entorno.

Por ello, surge la necesidad de diseñar otro tipo de redes que se adecuen a este tipo de entornos. Es entonces cuando se propuso el protocolo Bundle con el fin de obtener mejor rendimiento y fiabilidad en redes tolerantes al retardo y a las interrupciones (DTN) sustituyendo a la familia de protocolos de una red tradicional que no tenían un buen funcionamiento en este nuevo tipo de red.

En la actualidad, aunque se está realizando una gran investigación sobre el protocolo Bundle, sigue en fase de experimentación ya que tiene muchos aspectos en los que se puede mejorar. En este trabajo, se pretende dar a conocer las características del funcionamiento de una red tolerante al retardo y a las interrupciones y más concretamente del protocolo Bundle sobre el que funcionan. Para ello, después de ofrecer una visión teórica, se realizarán pruebas gracias al emulador de redes CORE, donde se podrá verificar cuál es su rendimiento en distintos escenarios y compararlos con el rendimiento del protocolo TCP de la red tradicional.

Palabras clave: red tolerante al retardo, red tolerante a interrupciones, protocolo Bundle, emulador CORE

ABSTRACT

Communication networks have acquired great importance over time due to the need of humans to have greater knowledge about everything that surrounds us. Currently, urban areas and rural areas in developed countries are well connected due to the traditional network of coaxial or fiber optic cable that is operating thanks to the TCP / IP protocol stack. However, there are still areas of our planet located for example in underdeveloped countries or in extreme environments such as large mountains or areas near volcanoes that also have the need of communication. In addition, traditional networks do not work well in communications between the Earth and outer space due to long distances and the environment.

Therefore, the need arises to design other types of networks that are suitable for this type of environments. It is when the Bundle protocol was proposed in order to obtain better performance and reliability in Delay and Disruption Tolerant Networks (DTN) replacing the family of traditional network protocols that did not have a good performance in this new type of network.

At present, although a great research is being carried out on the Bundle protocol, it is still in the experimentation phase since it has many aspects in which it can be improved. In this project, it is intended to present the characteristics and performance of Delay and Disruption Tolerant Networks and more specifically of the Bundle protocol on which they operate. To do this, after offering a theoretical vision, tests will be carried out thanks to the CORE network emulator, where we can verify its performance in different scenarios and compare it with the performance of the TCP protocol of the traditional network.

Keywords: delay tolerant network, disruption tolerant network, Bundle protocol, CORE emulator

AGRADECIMIENTOS

A Ignacio Soto, por su paciencia y ayuda continua en cualquier momento y durante el largo transcurso del proyecto.

A Lucía, por su apoyo incondicional y su motivación para poder terminar el trabajo desde que llegó a mi vida.

A Miguel, por su paciencia y consejos durante toda la carrera.

A mis padres y hermano, por estar siempre a mi lado tanto ahora como a lo largo de mi vida universitaria, siempre apoyándome.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ACRÓNIMOS

AES	*Advanced Exploration Systems*
CORE	*Common Open Research Emulator*
CCSDS	*Consultative Committee for Space Data Systems*
DDoS	*Denial of Service*
DINET	*Deep-Impact Networking experiment*
DSN	*Deep Space Network*
DTN	*Delay/Disruption Tolerant Network*
DTNRG	*Delay Tolerant Network Research Group*
DTNSEC	*Delay/Disruption Tolerant Network Security*
EID	*Endpoint Identifiers*
EO-1	*Earth Observing 1*
FIFO	*First In, First Out*
FTP	*File Transfer Protocol*
IAB	*Internet Architecture Board*
IANA	*Internet Assigned Numbers Authority*
ICN	*Intermittently Connected Network*
IETF	*Internet Engineering Task Force*
IP	*Internet Protocol*
IPN	*Interplanetary Network*
IRTF	*Internet Research Task Force*
LLCD	*Lunar Laser Communication Demonstration*
NASA	*National Aeronautics and Space Administration*
NEN	*Near Earth Network*
ONU	Organización de las Naciones Unidas
RFC	*Request for Comments*
SDNV	*Self-Delimiting Numeric Values*
SN	*Space Network*
SSD	*Space Services Department*
SSP	*Scheme-Specific Part*
TCP	*Transmission Control Protocol*

UDP	*User Datagram Protocol*
UIT	Unión Internacional de Telecomunicaciones
UNOOSA	*United Nations Office for Outer Space Affairs*
WLAN	*Wireless Local Area Network*

CAPÍTULO 1.

INTRODUCCIÓN

Las comunicaciones han sido una parte fundamental del ser humano desde su origen. En un principio se realizaba a través de la voz y diversas señas, sin embargo, por la necesidad de hacer más permanentes sus mensajes comenzaron a dibujar en cuevas para posteriormente dar nacimiento a la escritura. A lo largo de la historia más reciente, la necesidad de expandir las ideas por todo el planeta dio lugar a la aparición del telégrafo, el teléfono o el correo convencional. Una de las creaciones más importantes de la historia de las comunicaciones fue la aparición de Internet, que ha permitido al ser humano estar conectado desde cualquier punto del mundo a través de un conjunto de redes de comunicación interconectadas entre sí y que se comunican a través de unas normas establecidas en la familia de protocolos TCP/IP. Dicha familia de protocolos tiene un buen funcionamiento en escenarios en los que existe una conexión permanente entre los nodos, sin embargo, existen escenarios extremos en los que ha sido necesaria la creación de un nuevo protocolo. El protocolo Bundle ha sido desarrollado para lidiar con los desafíos existentes en este tipo de escenarios formados por redes tolerantes al retardo o las interrupciones. A lo largo de este trabajo de fin de grado se ofrecerá una mejor visión de este tipo de redes y del papel que realiza el protocolo Bundle dentro de ellas.

1.1 MOTIVACIÓN DEL TRABAJO

En la actualidad, vivimos en un mundo globalizado en el que la mayoría de las personas que lo habitan, sin tener en cuenta su localización geográfica, tienen la necesidad de comunicarse unas con otras. Además, por otro lado, el ser humano también siente la necesidad de conocer el espacio exterior que rodea a nuestro planeta y para ello es importante mantener un alto grado de fiabilidad en las comunicaciones con el exterior. Debido a estas razones, se ha producido un aumento paulatino en la aparición de las redes tolerantes al retardo y un crecimiento de su importancia en algunos ámbitos como el espacial ya que gracias al protocolo Bundle, puede obtener mejores rendimientos en algunas situaciones en las que la familia de protocolos TCP/IP no es efectivo. Por ello, para adquirir un mayor conocimiento sobre el protocolo y poder aprovechar al máximo

sus características es necesario realizar una investigación más exhaustiva. En este trabajo de fin de grado se realiza un estudio sobre las redes tolerantes al retardo y más concretamente sobre el protocolo Bundle y su funcionamiento respecto al protocolo TCP en algunos escenarios.

1.2 ESTRUCTURA DEL TRABAJO

El trabajo de fin de grado se encuentra dividido en 6 capítulos y un anexo que comprenden el siguiente contenido:

- Capítulo 1. Se lleva a cabo una introducción previa sobre cuál es el tema del trabajo, así como de la situación en la que se desarrolla tanto en el tema regulatorio como en el aspecto socioeconómico.
- Capítulo 2. Se describe el estado del arte realizando un estudio teórico sobre las redes tolerantes al retardo y las interrupciones y sobre el protocolo Bundle describiendo todas sus características.
- Capítulo 3. Se crea un entorno con el emulador de redes CORE en el que se generan distintos escenarios para representar algunas soluciones que puede aportar el protocolo Bundle respecto a TCP. Posteriormente se realiza una evaluación de los resultados obtenidos que nos permite comparar entre el protocolo Bundle y TCP.
- Capítulo 4. Se describe cual ha sido la planificación temporal del trabajo y el presupuesto que se ha dedicado para su realización.
- Capítulo 5. Se recogen las conclusiones obtenidas tras la realización del trabajo, así como de las posibles líneas sobre las que se podría investigar en un futuro.
- Anexo A. Se realiza un resumen más extenso del trabajo en inglés.
- Anexo B. Se analiza la configuración implementada en la realización de las pruebas.

1.3 MARCO REGULATORIO

El conjunto de protocolos de Internet se encuentra en continua evolución a través de la organización Internet Engineering Task Force (IETF) que se encarga de regular las propuestas y estándares de Internet mediante la publicación de documentos llamados

(por motivos históricos) Peticiones de Comentarios (RFC). Las RFC son publicadas después del diseño e implementación de los investigadores y tras la supervisión del Internet Architecture Board (IAB). Para publicar una RFC se deben seguir una serie de requisitos recogidos en la RFC 1543 [1]. En dicha RFC también se recogen los tres estados en los que se pueden encontrar, estándar, experimental o informativo. El protocolo Bundle se encuentra especificado en la RFC 5050 [2] donde se recogen sus características y también se indica que se encuentra en un estado experimental. El hecho de encontrarse en esta fase muestra que se requiere una mayor investigación para controlar completamente su funcionamiento y alcanzar el estado de estándar.

Una de las situaciones para las que está pensado el uso del protocolo Bundle es en las comunicaciones espaciales que se encuentran reguladas por la Unión Internacional de Telecomunicaciones (UIT). La UIT es un organismo especializado en las telecomunicaciones dentro de la Organización de las Naciones Unidas (ONU), que posee un departamento de Servicios Espaciales (SSD) encargado de coordinar los sistemas espaciales y las estaciones terrestres que se comunican con ellos. Dicho organismo trabaja en conjunto con la Oficina de Naciones Unidas para Asuntos del Espacio Exterior (UNOOSA) para llevar a cabo la publicación de la Ley Espacial [3] recientemente actualizada en 2017.

Como se puede observar, la importancia del marco regulador en este proyecto es baja al tratarse de un nuevo protocolo experimental que actualmente posee la capacidad de modificar sus especificaciones y parámetros para poder adecuarse a las situaciones y leyes de los ámbitos en los que sea útil su implementación. Además, al no realizar un despliegue del protocolo en un escenario real si no en un emulador de redes, se puede experimentar más allá de los límites del marco regulador con el fin de obtener mayor experiencia e información sobre el funcionamiento del protocolo.

1.4 ENTORNO SOCIOECONÓMICO

En los últimos años, debido a nuestra condición de seres sociales y al aumento de las tecnologías, el uso de las telecomunicaciones ha sufrido un gran crecimiento teniendo una gran influencia en el desarrollo de diferentes ámbitos de la vida humana.

Según el último informe de la Unión Internacional de las Telecomunicaciones (UIT), de diciembre de 2018 [4], el 51,2% de la población mundial tiene acceso a Internet. Sin embargo, como se observa en la figura 1, la mayor parte se encuentra en los países más desarrollados mientras que la población de los países en desarrollo o menos desarrollados, solo puede acceder con mayor coste y con menores prestaciones. Esto se produce ya que la arquitectura actual de Internet y sus protocolos se caracteriza por retardos muy pequeños y conectividad casi permanente, lo que no ocurre en zonas de difícil acceso o zonas rurales de muchos países en desarrollo.

Para hacer frente a esta situación, las redes inalámbricas se encuentran en plena investigación desarrollando nuevos protocolos y modelos de red que puedan ser desplegados en diversos entornos extremos en los que se puedan producir interrupciones en la red debido a las condiciones operativas. El desafío en la actualidad consiste en obtener una alta probabilidad de entrega correcta en el mensaje sin información de enrutamiento y sin una infraestructura fija de red.

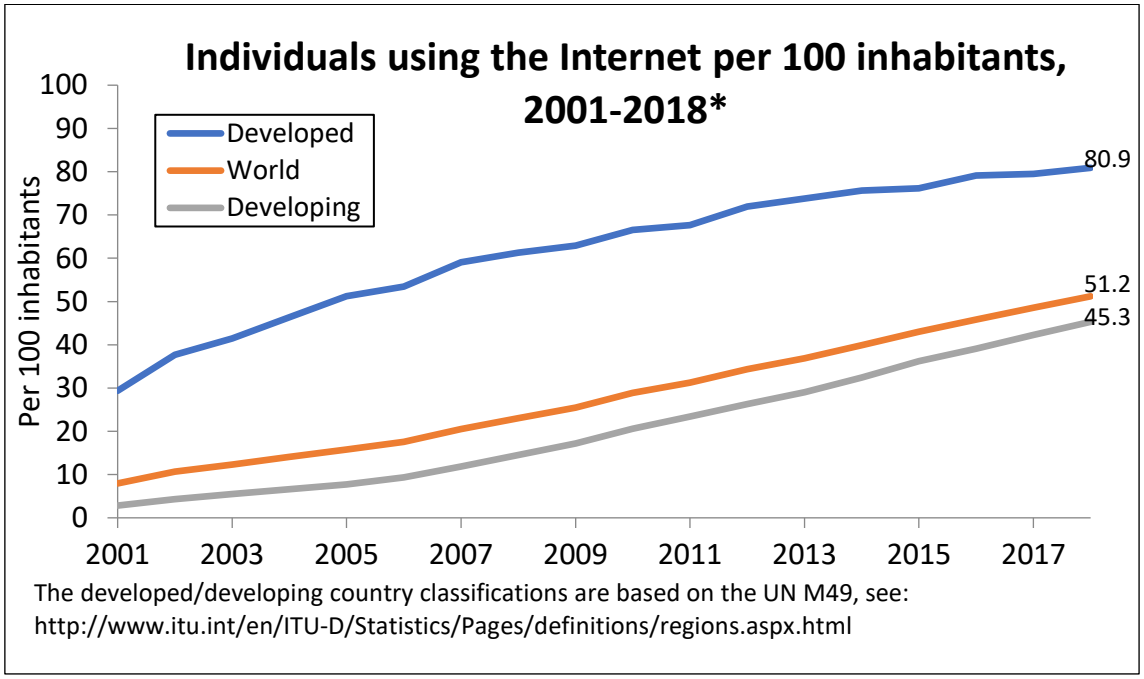

Figura 1. Uso de internet en el mundo
Fuente: [4]

Uno de los modelos de red propuestos son las redes de conexión de intermitente (ICN) que se basa en la cooperación entre distintos nodos para facilitar la comunicación. Este modelo no requiere de una infraestructura de red fija por lo que resulta muy atractivo. Concretamente las redes tolerantes al retardo y las interrupciones (DTN) se centran en

resolver los diversos problemas que poseen las ICNs y el protocolo Bundle, que se estudia en este trabajo, es una de las alternativas para implementarlas. Este nuevo protocolo que se encuentra en fase experimental para este tipo de redes podría tener un papel fundamental en la implantación de dichas redes en zonas en las que actualmente la conexión es prácticamente imposible de llevar a cabo. Para su correcta implementación, se deberían realizar acuerdos con los gobiernos de los distintos países, así como con los operadores de telecomunicaciones que trabajen allí, para desarrollar dispositivos que soporten este protocolo.

Por otro lado, el entorno del espacio exterior constituye otro ámbito que se encuentra en continua evolución y donde se podría aplicar este tipo de red con el protocolo Bundle. En este entorno se producen comunicaciones en las que la información es transmitida y recibida en un espacio de millones de kilómetros por lo que el retardo es muy alto, suelen existir interrupciones en la comunicación y el riesgo de pérdida de información en las comunicaciones es potencialmente alto. La agencia espacial más importante del mundo, es decir, la Administración Nacional de la Aeronáutica y del Espacio (NASA), ha tomado el modelo de red DTN como solución a estos problemas y lo implementó en junio de 2016 en la Estación Espacial Internacional [5]. Además también lo ha utilizado en alguna de sus misiones como “Deep Impact Networking” (DINET) [6] o “Lunar Laser Communication Demostration” (LLCD) [7].

Este trabajo tiene un propósito experimental con el fin de obtener más información sobre el funcionamiento y rendimiento del protocolo Bundle, pero no tiene como objetivo la explotación comercial directa. Sin embargo, teniendo en cuenta la importancia que puede adquirir el protocolo en un futuro cercano, los conocimientos adquiridos pueden ser útiles para realizar proyectos en los que se desarrollen productos basados en el protocolo Bundle con una aplicación comercial.

CAPÍTULO 2.

ESTADO DEL ARTE

En este capítulo se recogen los conceptos teóricos más importantes sobre las redes tolerantes al retardo y su protocolo de transporte, el protocolo Bundle.

2.1 REDES TOLERANTES AL RETARDO

2.1.1 Historia

Como se ha comprobado en los puntos anteriores, las DTN están adquiriendo una especial importancia en muchos campos actualmente.

Históricamente, las redes de datos han sido diseñadas teniendo en cuenta la existencia de al menos un camino extremo a extremo que garantizase el tránsito de información entre el origen y el destino. En estas redes se supone que cualquier enlace que conecte los dos nodos tiene que ser bidireccional y debe admitir una tasa de transmisión de datos simétrica, así como una baja probabilidad de error y latencia. Los nodos en este tipo de redes se encuentran en funcionamiento la mayor parte del tiempo y los cortes de energía en la red son poco frecuentes. En las redes tradicionales, los paquetes recibidos por los nodos intermedios son almacenados en búfer durante un corto periodo de tiempo hasta que se reenvíen al siguiente salto. El tamaño de los búferes es relativamente pequeño y se encuentran optimizados para mantener una baja tasa de pérdida de paquetes cuando se sobrecargan. Estos puntos conforman la base de Internet como una red global de conmutación de paquetes usando la torre de protocolos TCP/IP.

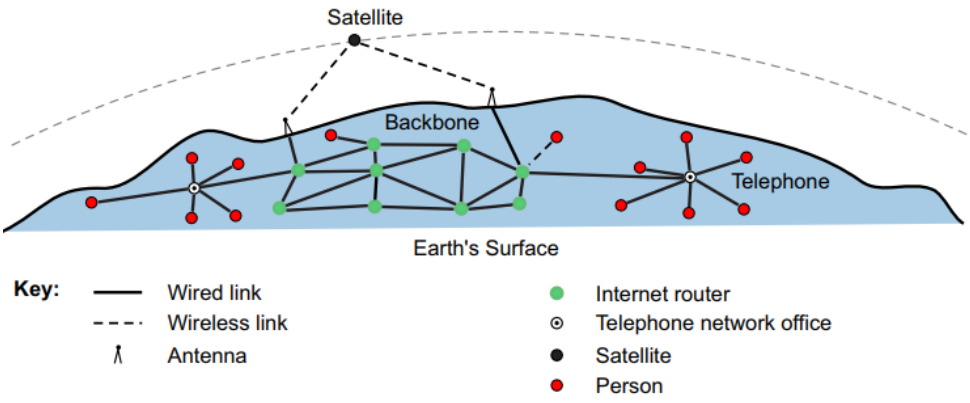

Figura 2. Arquitectura de una red tradicional
Fuente: [8]

Estas suposiciones son difíciles de conseguir cuando se modelan en la actualidad redes para ser desplegadas en entornos extremos (regiones volcánicas, espacio exterior, zonas rurales en desarrollo...) que sufren condiciones difíciles de superar imposibilitando el despliegue de la red tradicional. Estas situaciones están caracterizadas por una conectividad de red intermitente que no garantiza la existencia de una ruta extremo a extremo entre dos nodos debido a:

- Frecuentes cortes de energía.
- Alta movilidad de los nodos.
- Restricciones severas del ancho de banda.
- Retardos muy altos en los enlaces.
- Tamaño limitado del búfer tradicional que puede provocar sobrecargas y la consecuente pérdida de paquetes.

Por ejemplo, en las zonas rurales o en desarrollo, los cortes de energía provocan interrupciones temporales de la comunicación extremo a extremo de dos nodos. En el ámbito del espacio exterior, si un satélite se tiene que comunicar con una estación terrestre se producen retardos muy altos debido a las largas distancias que se deben cubrir. Además, también se pueden producir interrupciones en las comunicaciones si el satélite se encuentra orbitando por la parte oculta de otro planeta.

En los últimos 10 años se han realizado muchas investigaciones que han propuesto soluciones para superar estos problemas. Sin embargo, la mayoría de dichas investigaciones, han abordado el problema enfocándose en los protocolos de red tradicionales (TCP/IP) que en estos entornos no resultan útiles, por lo que finalmente aparecieron las redes tolerantes al retardo (DTN) con el protocolo Bundle como principal elemento de transmisión de datos.

El primer problema al que se enfrentaron los investigadores fue el de las comunicaciones interplanetarias en el espacio exterior ya que pretendían desarrollar una nueva arquitectura que fuese útil tanto en un entorno terrestre como interplanetario. Para resolverlo, los miembros del *Consultative Committee for Space Data Systems* [1](CCSDS)

[1] Comité formado por las principales agencias espaciales del mundo con el fin de proponer y desarrollar estándares para los datos espaciales y los sistemas de información

publicaron el protocolo de Entrega de Archivos (*File Delivery Protocol*) que hacía frente a los largos retardos en la propagación entre los nodos y las altas probabilidades de error. A medida que se desarrollaba este campo, las redes interplanetarias comenzaron a ser consideradas un subcampo dentro de las DTN. Debido a que las DTN afectan a un mayor número de usuarios y a que tiene un mayor interés tanto de forma académica como económica, la Internet Research Task Force (IRTF) creo el grupo de investigación de DTN (DTNRG) con muchos de los investigadores de las redes interplanetarias (IPN). Dicho grupo, propone en 2007 la arquitectura de red DTN y el protocolo Bundle a través de las RFC 4838 [9] y RFC 5050 [2] respectivamente.

Más tarde, en 2008 se propusieron los protocolos Saratoga [10] y el protocolo de transmisión Licklider [11] que son considerados protocolos de convergencia. Dichos protocolos se pueden usar entre el protocolo Bundle y TCP o UDP siendo un complemento que mejora las prestaciones de las comunicaciones en el enlace entre los nodos. El grupo DTNRG es quien documenta las especificaciones de estos nuevos protocolos que se muestran en las RFCs. Desde entonces, muchos investigadores han estado centrados en los diversos campos relacionados con DTN haciendo experimentos y preparando posibles despliegues en escenarios reales.

2.1.2 Características de una DTN y su solución

Una red tolerante al retardo y a las interrupciones (DTN) es una arquitectura de red formada por distintos nodos que a través del uso del protocolo Bundle, utilizan su capacidad de almacenamiento y mecanismos de replicación y reenvío entre otros, para poder establecer la comunicación superando las deficiencias existentes en entornos extremos. Las DTN se utilizan en entornos donde por su complejidad, las redes tradicionales no funcionan correctamente. Con DTN, en lugar de los enrutadores tradicionales, se utilizan nodos DTN que poseen una capacidad de almacenamiento muy superior. En una DTN, el enrutamiento se realiza mediante la técnica de almacenamiento y reenvío, donde la fuente o el nodo intermedio almacena el paquete cuando el enlace está inactivo hasta que vuelve a estar activo y se realiza el reenvío de datos al nodo más próximo. El grado de disponibilidad de la conexión ayuda a categorizar la red como si fuera DTN o no. A veces, incluso cuando el enlace está activo para un enrutador DTN, un paquete (*bundle*) puede perderse en la ruta, por lo que los

datos se pueden volver a retransmitir usando el protocolo Bundle o el protocolo TCP / IP estándar. En una DTN la comunicación es salto a salto, donde nodos los nodos sucesivos van transfiriendo la custodia del paquete hasta que llega a su destino. Si se pierde un paquete, y la custodia no es reconocida por el siguiente nodo, el último custodio debe retransmitir para mantener la mayor fiabilidad posible. Algunas de las propiedades que debe reunir un protocolo para tener un buen funcionamiento en una DTN son:

- Soporte de una conexión intermitente. Debido a que la energía de cada nodo es limitada y a que los nodos suelen encontrarse dispersos y poseen capacidad de movimiento, no se encuentran siempre visibles por lo que la red se desconecta frecuentemente dando lugar a cambios en la topología. La red puede permanecer inalcanzable durante largos periodos de tiempo por lo que los nodos cierran los enlaces periódicamente para conservar energía ya que no se puede garantizar un correcto funcionamiento de la red si se acaba la energía del nodo. En una DTN, la densidad de nodos es comparativamente menor y más distante que la de una red tradicional. La red es susceptible al cambio frecuente de ubicaciones de nodos con alta movilidad que se comunican solo cuando están en contacto con el rango de transmisión del otro.
- Asimilación de altos retardos y bajas eficiencias. El retardo extremo a extremo es la suma del retardo total de cada salto en la ruta. El retardo en cada salto está constituido por el tiempo de espera, el tiempo de espera en cola y el tiempo de transmisión. Cada retraso de salto puede ser muy alto ya que la comunicación puede tener un destino muy alejado del origen y al ser una conexión intermitente, la red se mantiene inalcanzable durante mucho tiempo y, por lo tanto, conduce a una velocidad de datos más baja mostrando características asimétricas en cuanto a tasa de subida y bajada de datos. Además, el retardo de espera en cola desempeña un papel principal en el retardo de extremo a extremo y las frecuentes fragmentaciones en DTN hacen que el retardo de espera en cola se incremente.
- Capacidad de almacenamiento persistente. Al soportar conexiones intermitentes, es necesario que los datos sean almacenados durante un periodo

indefinido de tiempo en el nodo. Se necesita este tipo de almacenamiento ya que, el siguiente nodo puede no estar disponible o los datos deben ser retransmitidos a otro nodo si se ha producido un error en la comunicación. Por otro lado, el espacio de almacenamiento limitado de un red tradicional conlleva una mayor tasa de pérdida de paquetes debido a una mayor probabilidad de saturación del almacenamiento.

- Fortalecer la seguridad. En general, las DTN son vulnerables a la suplantación de rutas, modificación de mensajes, ataques de denegación de servicio (DDoS) y otras amenazas de seguridad debido a que carecen de un servicio de seguridad especializado y a la escasa experimentación en un escenario real.

En DTN, al tener conexiones intermitentes, los nodos no se encuentran continuamente en línea. La comunicación en una DTN está sujeta a numerosas restricciones que varían en el tiempo. El momento en el que se lleva a cabo la comunicación entre dos nodos se denomina contacto y es posible que existan varios entre dos nodos, ya que por ejemplo un nodo puede tener varias opciones dependiendo el coste y la eficiencia del enlace. Según la RFC 4838 [9] hay 5 tipos de contactos:

- Oportunista. Es una conexión no programada que solo es posible que se establezca cuando el nodo que emite la información sabe que el nodo receptor está activo y puede recibir a información. Sucede por ejemplo cuando un avión no programado vuela mostrando su disponibilidad para la comunicación, y por casualidad, establece comunicación con un aeropuerto o con otro avión.
- Programado. Se crea un acuerdo para establecer el contacto a una hora determinada y por un tiempo determinado. Este tipo de contacto se establece cuando origen y destino siguen un patrón de movilidad como ocurre por ejemplo con un satélite alrededor de la Tierra.
- Pronosticado. Se trata de una conexión basada en una predicción creada por el número de contactos y duración del contacto que se ha establecido históricamente. Los recursos son reservados para este tipo de contacto y siempre se tiene en cuenta la probabilidad de que el contacto no tenga éxito o que no se establezca. Se encuentra en investigación.

- Persistente. Ambos nodos se encuentran siempre disponibles para establecer la comunicación. Es el tipo de contacto menos común en DTN ya que es más común en conexiones de Internet.
- Bajo demanda. Requiere la acción por parte de alguno de los nodos, para luego pasar a actuar como contacto persistente.

2.1.3 Tipos de nodos

Según la RFC 4838 [9], la arquitectura de una DTN está formada por regiones que son cada una de las redes que forman la DTN y que poseen unas características de comunicación homogéneas. Cada región está formada por nodos que se encargan de enviar o recibir (o ambas) paquetes utilizando el protocolo Bundle. Según el uso que hagan del protocolo, se distinguen 3 tipos de nodo:

- Host. Es un nodo que puede ser el origen o el destino final de la comunicación ya que envía o recibe los paquetes, pero sin reenviarlos. En este caso, el nodo requiere de una gran capacidad de almacenamiento persistente ya que pueden darse situaciones en las que existan largos retrasos hasta que se consigue disponibilidad en el enlace para transportar la información.
- Router. Este tipo de nodo se encarga de reenviar paquetes dentro de una región DTN y en algunas ocasiones pueden tener las funciones de host. Al igual que ocurre en el host, en este caso, la capa bundle debe soportar largos retrasos en algunas ocasiones por lo que se necesita una gran capacidad de almacenamiento permanente. En este tipo de nodos, el mecanismo de transferencia de custodia es opcional.
- Gateway. Son los nodos dedicados al reenvío de paquetes entre dos o más regiones DTN y también pueden hacer en algunas ocasiones las labores del host. En este caso, además de poseer una gran capacidad de almacenamiento permanente es muy recomendable la implementación del soporte para la transferencia de custodia. Gracias a los gateways, es posible la comunicación entre las capas que se encuentran por debajo de la capa Bundle y las regiones en las que trabajan.

2.1.4 Tipos de enrutamiento

Cuando se produce un contacto, ambos nodos intercambian un resumen de la información necesaria para actualizar su conocimiento sobre el entorno en el que opera. Por ejemplo, pueden intercambiar la lista de paquetes en la cola o la lista de nodos encontrados. Todas las acciones siguientes que se realizan en el nodo se basan en este conocimiento del modo que se observa en la figura 3. Dicho conocimiento es usado por la administración de colas (QM) para asignar una categoría a los paquetes en la cola. Si la cola está llena, la administración de colas aplica una política de descarte adecuada para dejar espacio a los paquetes que está recibiendo. Por otro lado, la política de reenvío (FW) selecciona los paquetes que se deben reenviar y que es probable que se dupliquen (R) para mejorar la efectividad del protocolo [12].

- Gestión de colas (QM). Define el orden total de los paquetes en la cola basándose en los conocimientos que posee el nodo. QM ordena todos los paquetes, incluso aquellos que no son candidatos para ser enviados. Cuando se deben añadir nuevos paquetes a la cola y no hay espacio, la política de descarte selecciona los paquetes de acuerdo con ese orden. En una DTN, los paquetes pueden estar en la cola durante mucho tiempo debido a la naturaleza de la red. Este hecho implica que, debido al tamaño limitado del búfer de los nodos DTN, se debe adoptar una política de administración de colas eficiente para evitar el descarte de paquetes importantes. Algunas de las principales políticas que se siguen son: FIFO (Primero en entrar, primero en salir), independiente del destino (no se tienen en cuenta parámetros relacionados con el destino) o dependiente del destino (se tienen en cuenta parámetros relacionados con el destino como el coste, la tolerancia a fallos…)
- Reenvío (FW). Se encarga de elegir el conjunto de paquetes de la cola que se deben reenviar. Normalmente los paquetes se seleccionan siguiendo el orden establecido por el QM, pero en algunos casos especiales se puede utilizar un nuevo orden para el reenvío. Dicho orden, es específico para cada contacto y dura solo el tiempo en el que se ha establecido el contacto. Existen tres tipos:
    - Basado en el conocimiento del nodo en términos de información contextual, (información sobre el estado del nodo en cuanto a nivel

batería, velocidad de movimiento, dirección…) información histórica, (información obtenida a lo largo del tiempo sobre la duración de los contactos, los nodos con los que se ha establecido contacto…) e información social (información que describe las relaciones entre los usuarios) que permiten predecir comportamientos futuros y mejorar la efectividad en el reenvío.

  - Contacto directo: el paquete se entrega directamente al destinatario sin nodos intermedios.
  - Permanente: siempre que se establece un contacto se realiza el reenvío de paquetes. Requiere un tiempo de computo muy bajo, pero se pueden realizar un alto número de transmisiones con las consecuencias en términos de congestión que ello conlleva.

- Replicación (R). Se encarga de controlar y limitar el número de copias de un paquete en la red. Gracias a la replicación, un mensaje que ha sido seleccionado por la política de FW para ser entregado al contacto actual, puede ser replicado y encolado nuevamente. De esta manera, si falla la comunicación con el contacto actual, se dispondrá de una copia que puede ser entregada de nuevo a la red. Existen varios tipos de replicación: copia única, (el paquete no se replica, una vez que se envía, se borra) limitada, (existen un número de réplicas del paquete limitadas) controlada (solo se realiza la replicación si se cumplen unas condiciones) e ilimitada (no hay restricción en el número de réplicas).

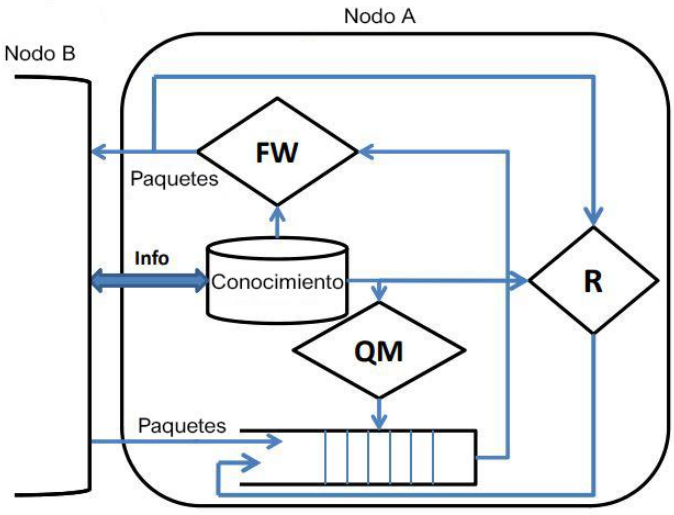

Figura 3. Acciones en el nodo
Fuente: [12]

Según el uso que hagan de las acciones anteriores, existirán los siguientes métodos de enrutamiento en una DTN[2]:

- (1) Entrega directa: mediante este método, los paquetes son enviados directamente al nodo final sin nodos intermedios y no requiere ningún conocimiento sobre la red. Es un enrutamiento con replicación de copia única, por lo que solo existe una copia del paquete en la red y tiene una gestión de colas del tipo FIFO.
- (2) Primer contacto: es el método de enrutamiento más simple. Con este método los paquetes se entregan cuando se establece el primer contacto y se eliminan de la cola. Solo existe una copia del paquete en la red (copia única) y usa colas del tipo FIFO.
- (3) Epidémico: es un protocolo basado en la inundación por el que los paquetes son transmitidos a todos los nodos vecinos sin tomar ninguna decisión en el enrutamiento. Cuando no hay hueco en la cola, se descartan los paquetes más antiguos. Las réplicas no tienen ningún límite (ilimitadas) y el reenvío es según la política FIFO. Con este protocolo se garantiza una tasa de entrega máxima, a cambio de la posibilidad de generar una sobrecarga de paquetes en algunos nodos que reduzca el rendimiento y la escalabilidad de la red. La sobrecarga de paquetes se produce ya que el mensaje puede continuar propagándose incluso una vez que ya ha llegado al nodo final, por lo que este protocolo se usa si no es posible emplear un método de enrutamiento más efectivo.
- (4) Pulverización y espera (Spray and Wait): este protocolo consta de dos fases. En la fase de pulverización el nodo de origen genera L (replicación limitada) copias de los paquetes y los distribuye a L nodos distintos de manera que cada nodo tiene una copia del mensaje. El nodo que posee la copia del paquete entra en la etapa de espera almacenando el paquete recibido hasta que encuentra el nodo de destino al que se lo tiene que entregar. Mediante este protocolo se combina la velocidad del enrutamiento epidémico con la velocidad y simplicidad de la transmisión directa. La política de reenvío es una mezcla de reenvío

[2] En la tabla 1 se resumen las características de los protocolos

permanente (pulverización) y contacto directo (espera) y la administración de las colas es FIFO.

Existe una variante del protocolo original que es la pulverización y espera binaria en la que el nodo de origen envía *L/2* copias del paquete al primer nodo encontrado. El nodo que las recibe vuelve a enviar la mitad de las copias del paquete recibido hasta que solo queda una copia y entra en la fase de espera como en el protocolo original.

- (5) Pulverización difusa: se trata de un método basado en dos parámetros internos, el tamaño del mensaje y el "Forward Transmission Count" (FTC). Ambos parámetros sirven para crear una regla que prioriza los paquetes que deben ser transmitidos a todos los vecinos. En este método, la probabilidad de sobrecarga es muy inferior en comparación con otros protocolos. La gestión de colas en este caso es independiente del destino y se basa en la prioridad que se ha establecido para el mensaje. Las replicaciones del paquete son ilimitadas y utiliza la técnica de reenvío permanente.
- (6) PRoPHET: utiliza la predicción como forma de evaluar la probabilidad de que un nodo consiga enviar un paquete que alcance con éxito su destino. Según este método, el nodo de origen envía una copia del paquete a todos los nodos vecinos que tenga una probabilidad de entrega mayor que la suya. Utiliza una administración de colas FIFO y la replicación de paquetes es ilimitada. La política de reenvío es basada en la información histórica.
- (7) SCAR: con este método, al igual que ocurre en PRoPHET, los paquetes son enviados al nodo vecino que tiene mayor probabilidad de entrega al destino. En este caso, para calcular la probabilidad de reenvío se basa en la información histórica y contextual, en la colocación, en la movilidad y en el nivel de batería. Para cada paquete existe un paquete maestro y L réplicas de respaldo que son generadas en el origen. Los paquetes son ordenados en la cola independientemente del destino de forma que los paquetes maestros son enviados primero y las copias de respaldo pueden ser descartadas si es necesario (los paquetes maestros no).
- (8) FAD: es un método similar a PRoPHET, pero en este caso, los mensajes son ordenados en la cola basándose en la tolerancia a fallos. La tolerancia a fallos es

proporcional al número de réplicas que hay en la red y también tiene en cuenta la probabilidad de entrega. Los paquetes con menor tolerancia a fallos son reenviados antes para aumentar de forma ilimitada el número de copias del paquete que hay en la red y de esta forma aumentar la probabilidad de ser entregados con éxito. En este método, el reenvío está basado en la información histórica y la gestión de colas es dependiente del destino.

- (9) MaxProp: con este método, cada nodo tiene una tabla de enrutamiento con la que conoce el coste de alcanzar otro nodo a través de sus vecinos actuales. Las tablas de enrutamiento se actualizan gracias a la información histórica recibida de los nodos vecinos y los paquetes son ordenados en la cola y reenviados basándose en un menor coste para llegar al destino. La replicación es ilimitada y la gestión de cola es una combinación de destino independiente ya que se conoce el número actual de saltos que el paquete lleva acumulados y de destino dependiente al utilizar el algoritmo de Dijkstra para saber el coste de llegar al destino.
- (10) RAPID: este método trata el enrutamiento como un problema de asignación de recursos y por ello trata de optimizar una métrica de enrutamiento concreta como puede ser el retraso en la entrega asignando un grado de utilidad a cada paquete. La gestión de colas es dependiente del destino ya que los paquetes son ordenados según su utilidad. Con este método, cuando el búfer está completo, los paquetes con menor utilidad se eliminan. La técnica de reenvío está basada en la información histórica y se hacen réplicas de forma ilimitada.
- (11) Basado en clústeres: la base de este protocolo es la agrupación de los nodos que posean una probabilidad de entrega similar en un clúster. Dentro del clúster, los nodos pueden intercambiar sus recursos de forma que se reduzcan los gastos generales y se equilibre la carga. En este protocolo, la replicación es de copia única, la gestión de cola es FIFO y la técnica de reenvío está basada en la información histórica.
- (12) NECTAR: este método utiliza reenvío basado en el historial de contactos que ha tenido el nodo con el que genera un índice de vecindad. La replicación de mensajes es controlada mediante parámetros que provienen del contexto del nodo y la gestión de cola es independiente del destino.

- (13) ORWAR: este protocolo utiliza la información contextual del nodo como la velocidad, ancho de banda, dirección del movimiento… para estimar la duración del contacto y solo se reenviarán los paquetes que puedan transmitirse dentro de ese intervalo. Este protocolo genera una función llamada relación de utilidad por bit, que permite ordenar los paquetes en la cola independientemente del destino y controlar la cantidad de réplicas de cada paquete.
- (14) HiBOp: en este método solo se reenvían los paquetes a los nodos con mayor probabilidad de entrega. La probabilidad de entrega se basa en la información histórica y contextual del nodo que también sirve para replicar los paquetes de forma controlada. Solo el nodo inicial de origen puede replicar el paquete, el resto solo puede reenviarlo. La gestión de cola es tipo FIFO.
- (15) BubbleRap: es un protocolo basado en la información social. Con BubbleRap la gestión del reenvío se basa en el conocimiento sobre la estructura de la red y sobre la centralidad de los nodos. Cada paquete se clasifica según dos parámetros, global y local. El reenvío se realiza siguiendo la clasificación global hasta que el paquete llega a un nodo que se encuentra en la misma comunidad que el nodo de destino. En este momento, se utiliza la clasificación local para llegar al destino. Los nodos que reenvían los paquetes no pueden descartar los paquetes hasta que no ha llegado a la comunidad del nodo de destino. En este caso la replicación es ilimitada y la gestión de cola es FIFO.
- (16) SimBet: al igual que BubbleRap es un protocolo basado en la información social. El reenvío está basado en la función de utilidad que se calcula según los parámetros de similitud e intermediación. En este caso, solo existe una única copia del mensaje en la red y la gestión de colas es FIFO.

		PROTOCOLOS															
		(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)	(15)	(16)
FW	Contac. Direct.	✔	✕	✕		✕	✕	✕	✕	✕	✕	✕	✕	✕	✕	✕	✕
	Permanente	✕	✔	✔		✔	✕	✕	✕	✕	✕	✕	✕	✕	✕	✕	✕
	Contexto	✕	✕	✕	✕	✕	✕	✔	✕	✕	✕	✕	✕	✔	✔	✕	✕
	Histórico	✕	✕	✕	✕	✕	✔	✔	✔	✔	✔	✔	✔	✕	✔	✕	✕
	Social	✕	✕	✕	✕	✕	✕	✕	✕	✕	✕	✕	✕	✕	✕	✔	✔
R	Única	✔	✔	✕	✕	✕	✕	✕	✕	✕	✕	✔	✕	✕	✕	✕	✔
	Limitada	✕	✕	✕		✕	✕	✔	✕	✕	✕	✕	✕	✕	✕	✕	✕
	Ilimitada	✕	✕	✔	✕	✔	✔	✕	✔	✔	✔	✕	✕	✕	✕	✕	✕
	Controlada	✕	✕	✕	✕	✕	✕	✕	✕	✕	✕	✕	✔	✔	✔	✔	✕
QM	FIFO	✔	✔	✔	✔	✕	✔	✕	✕	✕	✕	✔	✕	✕	✔	✔	✔
	Dest. Dep.	✕	✕	✕	✕	✕	✕	✕	✔	✔	✔	✕	✕	✕	✕	✕	✕
	Dest. Ind.	✕	✕	✕	✕	✔	✕	✔	✕	✔	✕	✕	✔	✔	✕	✕	✕

Tabla 1. Tabla de características de los protocolos de enrutamiento

2.1.5 Seguridad

En una DTN, al tener unas condiciones especiales, muchos de los protocolos que funcionan con efectividad en redes no tolerantes al retardo, no son aplicables. En las redes tolerantes al retardo, se dispone de una cantidad de recursos limitada y por ello es aún más necesaria la existencia de un mecanismo de autenticación y control de acceso que en una red tradicional. El acceso a la red por parte de un usuario que no esté autorizado podría generar tráfico que consuma los recursos que podría necesitar un usuario autorizado. Para ello se establecen los siguientes objetivos básicos de seguridad en DTN reflejados en la RFC4838 [9]:

- Evitar que existan datos en la DTN que pertenezcan a aplicaciones no autorizadas o que dichas aplicaciones ejerzan algún tipo de control sobre la infraestructura de la DTN.
- Evitar que las aplicaciones que están autorizadas para un modo de envío de paquetes utilicen la red para otro tipo de servicios para los que carecen de permiso.
- Eliminación inmediata de los paquetes dañados o modificados incorrectamente durante el transporte.
- Detectar de inmediato y desautorizar aplicaciones que no cumplan con los requisitos de seguridad.

Siguiendo los objetivos establecidos, los nodos DTN descartarán el tráfico inmediatamente si han fallado los controles de autenticación y acceso. Este hecho hace que los ataques de denegación de servicio (DDoS) tengan poca efectividad en este tipo de redes. Para intentar cumplir estos objetivos, en DTN se adopta una arquitectura de seguridad estándar que se encuentra opcionalmente implementada llamada DTNSEC [13]. DTNSEC utiliza mecanismos de autenticación en cada nodo para poder manejar el control de acceso al nodo tanto para el reenvío como para el almacenamiento de datos y, por otro lado, mecanismos para asegurar la integridad de la capa de aplicación. Sin embargo, la existencia de dichos mecanismos conlleva una sobrecarga de almacenamiento de credenciales y un mayor tiempo de cómputo que implican una pérdida de eficiencia.

2.1.6 Aplicaciones

La aplicación de la arquitectura DTN ha ido creciendo a lo largo del tiempo en dos grandes entornos principalmente:

- Redes interplanetarias: fue el primer campo donde se utilizaron las DTN debido a las características especiales del entorno ya que los protocolos estándar no funcionaban correctamente en este ámbito. Las comunicaciones entre la Tierra y cualquier nave espacial son muy difíciles debido a las largas distancias, por ello la principal agencia espacial, la Administración Nacional de la Aeronáutica y del Espacio (NASA), ha adoptado la DTN como el método de interconexión más fiable en sus misiones [14]. Para comunicarse a través de estas largas distancias, la NASA utiliza tres redes de comunicación formadas por estaciones terrestres y satélites espaciales para la transmisión y recepción de datos que soportan misiones tanto de la NASA como de fuera de la NASA. Dichas redes son la Red de Espacio Profundo (DSN), la Red Cercana a la Tierra (NEN) y la Red Espacial (SN). El objetivo de la NASA es crear un Internet del Sistema Solar (SSI) [15] que funcione igual que en la Tierra utilizando los protocolos de DTN. La investigación en la NASA está siendo dirigida por el Proyecto de Sistemas de Exploración Avanzada (AES) apoyado por el Comité Consultivo para Sistemas de Datos Espaciales (CCSDS) y por el IETF.

En mayo de 2016, la Estación Espacial Internacional implementó un sistema con DTN que permitía mejorar la fiabilidad en las transmisiones de datos, reduciendo la sobrecarga y proporcionando una arquitectura para soportar misiones espaciales. Algunos de los proyectos de la NASA utilizando DTNs con el protocolo Bundle son:

- Misión de Observación de la Tierra 1 (EO-1): se llevó a cabo en 2011 cuando se introdujo la tecnología de las redes DTN, concretamente el protocolo Bundle, para realizar observaciones de la Tierra y para misiones de la órbita baja terrestre [16]. Para este proyecto, utilizaron un satélite llamado EO-1 que enviaba datos hacia la estación terrestre de la NASA en Wallops (Estados Unidos), hacia el Centro de Operaciones de la misión (MOC) y hacia el Centro de Operaciones de Ciencias (SOC) como se observa en la figura 4. La realización del proyecto se llevó a cabo en tres fases.

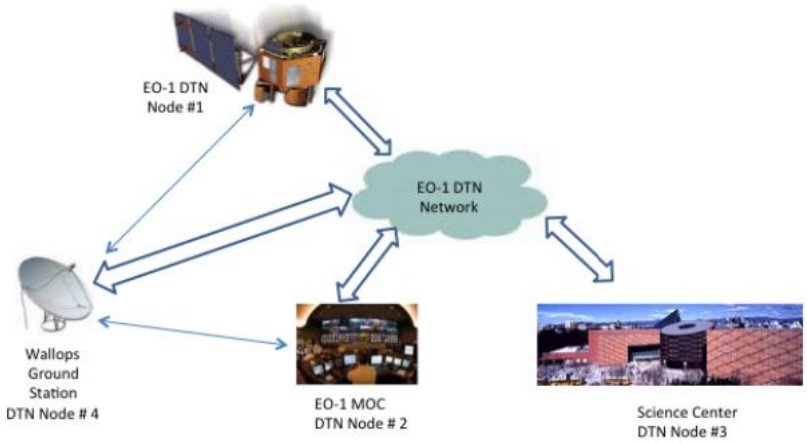

Figura 4. Escenario de la misión de Observación de la Tierra 1
Fuente: [16]

En la primera fase, se realizó una verificación de los enlaces de comunicación entre el satélite y el MOC permitiendo comprobar los beneficios de la tecnología DTN, ya que, sin su utilización, la transferencia de información hubiese sido incompleta. En la segunda fase, demostraron los beneficios de la capacidad de almacenamiento y reenvío introduciendo el nodo del SOC. Finalmente, en la tercera fase se

incorporó el nodo de la base terrestre de Wallops con el que pudieron comprobar el funcionamiento del método de transferencia de custodia.

- Demostración de Comunicación Lunar Láser (LLCD): en 2013, cuando se comenzó este proyecto, la NASA consiguió transmitir una señal óptica desde la Luna a la base terrestre de White Sands (Estados Unidos) a 622 megabits por segundo. Posteriormente, introdujeron el protocolo Bundle para realizar la comunicación con una mayor fiabilidad [7].

- Redes terrestres: aunque el entorno del espacio exterior fue el más importante en los inicios de las DTN, con el tiempo el entorno terrestre ha ido adquiriendo más importancia y por ello las DTN se han desarrollado en distintos aspectos:
  - Redes militares: en este ámbito, las DTN se enfocan en resolver los desafíos presentes en las redes militares como pueden ser, la alta movilidad de los nodos ya que no existe una infraestructura fija, la existencia de interferencias ambientales, la disponibilidad limitada del espectro, la introducción de interferencias por parte del enemigo y velocidades de datos reducidas y con larga latencia. En este entorno, se encuentra en desarrollo el proyecto CONDOR [17] de la infantería de marina de Estados Unidos en que el que se pretenden extender y unir redes de datos tácticas que hasta ahora se encontraban separadas por características del terreno o por la distancia. Como se puede observar en la figura 5, el desarrollo del proyecto se realiza a través de tres tipos de vehículos. El vehículo Gateway se utiliza para extender las comunicaciones más allá de la línea de visión a través de un satélite. El vehículo de Punto de Presencia (PoP-V) conecta radios o sistemas satelitales actuando como traductor y repetidor. El vehículo Jump C2 actúa como un puesto de comando móvil al mantener comunicaciones satelitales continuas y conectar vehículos de comando cercanos mediante tecnología inalámbrica. Sin embargo, en este entorno se debe mejorar la seguridad de la red ya que los datos que se manejan son muy delicados y están trabajando junto al DTNRG para desarrollar extensiones del protocolo Bundle que permitan una entrega fiable en multidifusión.

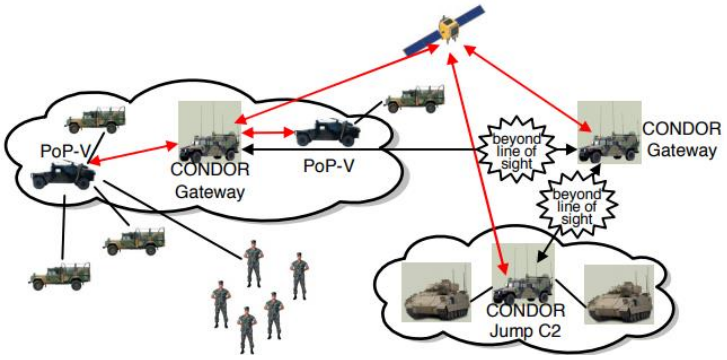

Figura 5. Red proyecto CONDOR
Fuente: [17]

- Redes para el control de la naturaleza: en este entorno, se están desarrollando numerosos proyectos, en favor del medio ambiente. Algunos de los proyectos más importantes que se están desarrollando son:
  - ZebraNet: es un proyecto de monitorización de vida silvestre que se lleva a cabo en la reserva natural Sweetwaters en Kenia [18]. Consiste en instalar a las cebras un collar con un sistema de posicionamiento global (GPS) para estudiar sus hábitos y comportamientos. Cuando la distancia entre dos collares está en el rango de comunicación, estos intercambiarían información mediante un algoritmo de enrutamiento epidémico. El experimento adicional de este proyecto es resolver los problemas de energía de los equipos, la adaptabilidad y la compresión de los datos.
  - SWIM: es un proyecto similar a ZebraNet en el que se monitoriza la actividad de las ballenas en los océanos [19]. Las ballenas controladas poseen una etiqueta que realiza el intercambio de información cuando se encuentra con otra ballena y a su vez con boyas situadas en los océanos que es donde se recopila toda la información.
  - Monitorización de la calidad de los lagos en Europa: desde la Unión Europea se ha aconsejado a los gobiernos estatales que realicen acciones para proteger la calidad del agua en los lagos. Para ello, en vez de usar la red tradicional, se usa DTN ya que tiene unos costes indirectos más bajos. Su funcionamiento consiste en emplear un nodo que navega por el lago actuando como almacén de datos y

cuando el nodo llega al puerto entrega la información a otros nodos que tienen conexión a internet [20].

- ❖ Monitorización ambiental en áreas metropolitanas (EMMA): se trata de un proyecto que tiene como objetivo controlar la contaminación en áreas metropolitanas desarrollando una arquitectura descentralizada y rentable para la medición del aire en todo el área [21]. La toma mediciones se realiza de forma continuada en varios sectores urbanos mediante vehículos de transporte público como autobuses y tranvías. Los valores obtenidos son intercambiados mediante WLAN, pero dado que los vehículos se encuentran de forma esporádica, han introducido las DTN. En este caso, la DTN, sirve para reducir los costes de transmisión que aparecen en las conexiones de datos móviles. Además de para distribuir valores de medición, la arquitectura de EMMA también se puede usar para, por ejemplo, el intercambio de información de pasajeros o saber cuándo realizar labores de mantenimiento en el vehículo.

Por otro lado, también existen proyectos enfocados a obtener una respuesta más efectiva y rápida frente a desastres naturales gracias a la monitorización sísmica y al control de los incendios.

- o Redes en zonas rurales, subdesarrolladas o en entornos remotos: con el desarrollo de las tecnologías de comunicación, los móviles e internet se han convertido en una parte importante de la vida social de las personas como medio de información y comunicación. Sin embargo, en muchas zonas alejadas de las grandes ciudades o situadas en países subdesarrollados, la falta de conexión a Internet, entre otras razones, ha impedido el desarrollo de la economía y cultura local ya que requiere de una arquitectura muy costosa. Para mejorar esta situación se han desarrollado varios proyectos entre los que están:

- DakNet: se trata de un proyecto que posee una estructura similar a la que se observa en la figura 6 y que tiene como objetivo proporcionar conectividad a aldeas rurales en India y Camboya [22]. En este proyecto, se instalan algunos sistemas informáticos básicos

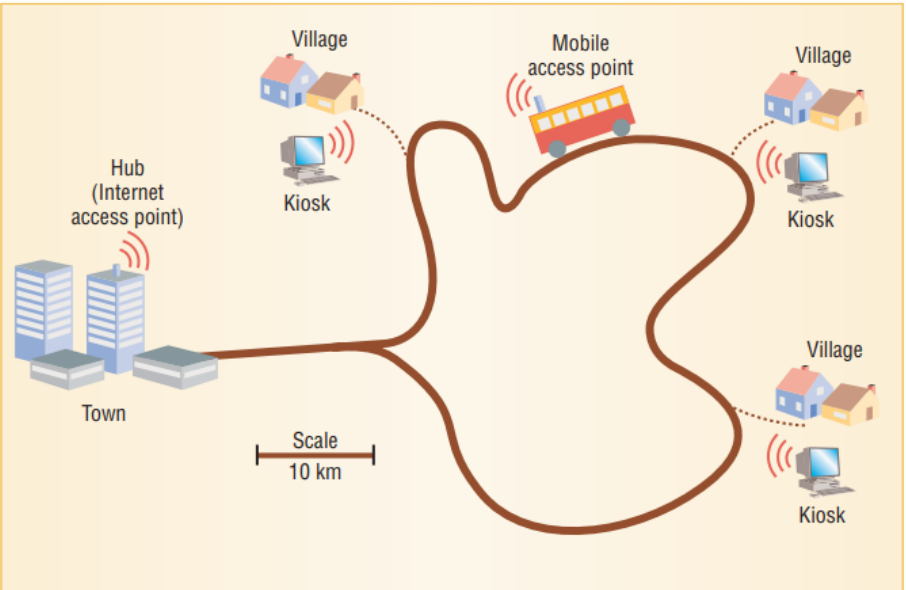

Figura 6. Red proyecto DakNet
Fuente: [22]

en pequeños stands con puntos de acceso para servir a los aldeanos que requieren acceso a correo electrónico, servicios bancarios, servicios gubernamentales, etc. Todas las solicitudes se almacenan en búfer en el punto de acceso y se emiten de forma inalámbrica oportunamente a cualquier vehículo habilitado para la conexión que pasa cerca del stand. A su vez, estos vehículos transferirán las solicitudes de los aldeanos a la ciudad más cercana donde se intercambian a través de Internet. Mediante este proyecto se consigue mantener unos bajos costos operativos y de configuración.

- Wizzy Digital Courier: es un proyecto similar a DakNet, desarrollado entre algunas escuelas en aldeas de Sudáfrica. Consiste en una serie de mensajeros que transportan un dispositivo de almacenamiento USB almacenando los datos que recibe de las escuelas y haciendo viajes periódicos a las ciudades que tienen conexión a internet para realizar el intercambio de información.

- N4C: es un proyecto desarrollado por diversos países entre los que se encuentra España. Tiene su aplicación en zonas remotas de Laponia y de la región montañosa de Kočevje (Eslovenia) y una estructura que se puede observar en la figura 7 [23]. Entre sus objetivos está el rastreo de renos, el uso por parte de los excursionistas y obtener datos meteorológicos y ambientales.

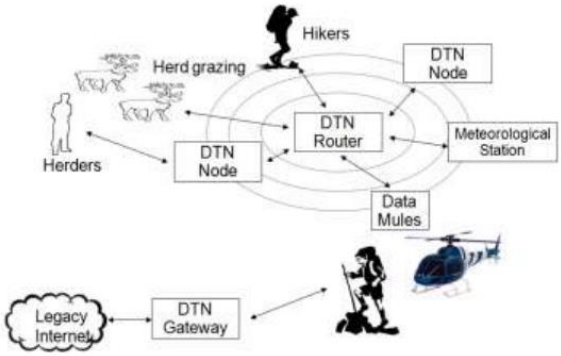

Figura 7. Red proyecto N4C
Fuente: [23]

- Redes vehiculares: este tipo de red surgió como un medio para mejorar la seguridad del tráfico y reducir el número de accidentes y su funcionamiento se puede observar en la figura 8. Las redes vehiculares están formadas dos tipos de nodos, estacionario y móvil [24]. Las estaciones de retransmisión estacionarias (SRS) se encuentran a lo largo de carreteras y autopistas. Muy pocas de las SRS (llamadas pasarelas en este caso) poseen una conexión a Internet y todas las demás están aisladas y, a menudo, muy separadas por lo que no pueden comunicarse directamente entre ellas. Por otro lado, se encuentran los nodos móviles montados sobre vehículos que discurren por las carreteras y que sirven como dispositivos oportunistas de almacenamiento y transferencia conectando con cualquier SRS. En este escenario debido a los diferentes patrones de conectividad a causa de la diferencia entre vehículos (camiones, coches, motos…) es necesario trabajar con tiempos de retraso y tolerancia a interrupciones. La comunicación vehicular también ha recibido particular atención para difundir información dependiente de la ubicación (por ejemplo, congestión de tráfico, disponibilidad de aparcamiento, etc.), así como proporcionar conectividad

básica a los pasajeros que viajan a través de los diversos vehículos de transporte.

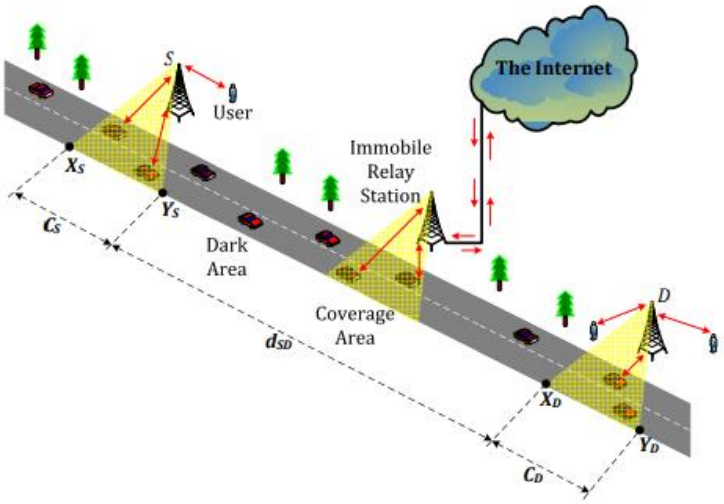

Figura 8. Red vehicular
Fuente: [24]

2.2 PROTOCOLO BUNDLE

2.2.1 Concepto

El protocolo Bundle es el protocolo principal definido por el IRTF para las redes DTN. Con el protocolo Bundle, toda la información es empaquetada en una sola entidad que recibe el nombre de bundle y es transmitido a través de la DTN. Su mecanismo de almacenamiento y reenvío y la opción de transferencia de custodia, le hacen el protocolo de entrega de datos más fiable en un entorno de comunicaciones extremo. Sin embargo, el protocolo Bundle no es capaz por sí mismo de realizar la comunicación ya que para poder establecer el intercambio de datos, necesita establecer contacto con los protocolos de las capas inferiores a través de diferentes tipos de adaptadores de capa de convergencia (CLA). Desde la perspectiva de la pila de protocolos estándar, como se observa en la figura 9, el protocolo bundle se sitúa en la capa que recibe el mismo nombre y permite la interoperación entre redes muy heterogéneas.

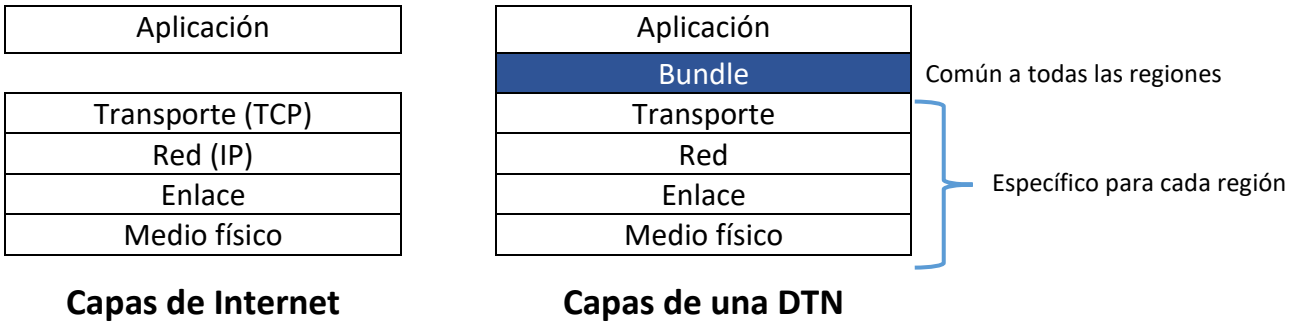

Figura 9. Comparación de capas en Internet y en DTN

2.2.2 Nombramiento de los nodos

Para la interoperabilidad entre los distintos nodos de la DTN, se utiliza un esquema de nombres flexible capaz de encapsular diferentes esquemas de nombres y números en la misma sintaxis de nomenclatura general. Cada nodo o grupo de nodos son nombrados como *End Point Identifiers* (EID) con un tamaño máximo de 1024 bytes y de estilo similar a un *Uniform Resource Identifier* (URI). Cuando el EID se refiere a más de un nodo, estamos en una situación de multidifusión. El EID según la RFC5050 [2] que define al protocolo Bundle, tiene la siguiente estructura:

< nombre del esquema > : < parte específica del esquema (SSP) >

En este caso, el nombre del esquema seria dtn (designado globalmente por la IANA). El traspaso de información sobre los EIDs hacia capas más bajas se realiza cuando se ha decidido a que nodo enviar la información. Este traspaso de información requiere un mecanismo de traducción que en el caso de la arquitectura de una DTN está basado en el principio de traducción tardía (*late binding*). Gracias a este principio, se puede identificar la parte específica del esquema (SSP) sin que sea necesario hacerlo en el origen, sino que se realiza cuando el paquete llega al destino. La traducción tardía resulta muy útil en este tipo de redes DTN donde los nodos tienen gran movilidad.

2.2.3 Formato del protocolo

Los paquetes generados por el protocolo Bundle deben ser una unión de al menos dos estructuras de bloques. Un bloque de cabecera primario obligatorio, un bloque de carga útil (*payload*) opcional y un conjunto de bloques de extensión opcionales que seguirán al bloque primario como el protocolo de seguridad de Bundle (BSP). El último bloque de la secuencia debe tener el bit indicador de último bloque a 1 para que el nodo lo

reconozca como último. Según la RFC5050 [2] el formato del bloque primario del paquete (bundle) se puede observar en la tabla 2.

Version	Processing Flags	Class of Service Flags	SRR Flags
Block Length			
Destination Scheme Offset		Destination SSP Offset	
Source Scheme Offset		Source SSP Offset	
Report-To Scheme Offset		Report-To SSP Offset	
Custodian Scheme Offset		Custodian SSP Offset	
Creation Timestamp (8 bytes)			
Lifetime (4 bytes)			
Dictionary Length			
Dictionary Array			
Fragment Offset			
Total Application PDU Length			

Tabla 2. Cabecera de un paquete del protocolo Bundle.

Los campos más importantes del bloque primario son:

- Version: indica la versión del protocolo en 1 byte.
- Bundle Processing Control Flags: es un conjunto de bits del 0 al 20 donde se muestra por ejemplo si el paquete está fragmentado o si se debe implementar el mecanismo de transferencia de custodia.
- Block Length: contiene la suma de la longitud de todos los campos restantes del bloque.
- Creation Timestamp: tiene un tamaño de 8 bytes e indica el momento de creación del paquete expresado en segundos desde el 1 de enero del año 2000 en la escala UTC. Junto con el EID de origen, la longitud de la carga útil (*payload*) y el desplazamiento del fragmento (si se ha realizado fragmentación) sirve para identificar el paquete.
- Lifetime: muestra el número de segundos posteriores al tiempo de creación durante los que el paquete será útil. Si se han superado los segundos que aparecen en este campo, el nodo podrá eliminar el paquete de la red, borrándolo de su almacenamiento. Tiene un tamaño de 4 bytes.
- Dictionary Length: contiene la longitud del diccionario.
- Dictionary: es una matriz de bytes resultado de la unión de los SSP que han sido referenciados anteriormente en algún campo de la cabecera.

- Fragment Offset: indica el desplazamiento del fragmento respecto al paquete original. Este campo solo aparece si en el campo "*Bundle Proccesing Control Flags*" se indica la existencia de fragmentación.
- Total Application Data Unit Length: indica el tamaño del paquete original, al que se ha realizado la fragmentación. Este campo solo aparece si en el campo "*Bundle Proccesing Control Flags*" se indica la existencia de fragmentación.

Algunos campos del bloque tienen un tamaño variable ya que hacen uso un esquema de codificación llamado *Self-Delimiting Numeric Values* (SDNV), cuyo funcionamiento se recoge en la RFC 6256 [25]. El uso de este esquema de codificación permite al protocolo Bundle hacer un consumo mínimo del ancho de banda de transmisión y permite flexibilidad a la hora de afrontar necesidades que puedan surgir en el futuro.

2.2.4 Almacenamiento y reenvío

En lugar de funcionar como una conexión extremo a extremo como los protocolos de transporte de Internet, la capa Bundle es un sistema de conmutación de paquetes que consiste en el almacenamiento, transporte y reenvío de los paquetes. Los paquetes o fragmentos de dichos paquetes se envían desde la unidad de almacenamiento de un nodo a la unidad de almacenamiento de otro nodo, a través de una ruta que debe llegar al nodo de destino final. Algunos los problemas relacionados con una DTN como la conexión intermitente, los largos retrasos o la tasa de envío asimétrica son subsanados en parte gracias a este mecanismo de almacenamiento y reenvío que se puede observar en la figura 10. Con este mecanismo, la transferencia de datos que se esté llevando a cabo, se puede pausar durante un largo periodo de tiempo si el enlace no está disponible ya que la información se mantendrá almacenada en el nodo de origen o en uno intermedio. Para hacer posible el almacenamiento de datos durante un periodo de tiempo impredecible (ya que en algunos casos no es posible saber cuándo se podrá volver a realizar el contacto entre dos nodos) el protocolo Bundle utiliza un almacenamiento de memoria persistente en lugar de la memoria volátil que se utiliza en Internet que solo permite el almacenamiento durante milisegundos.

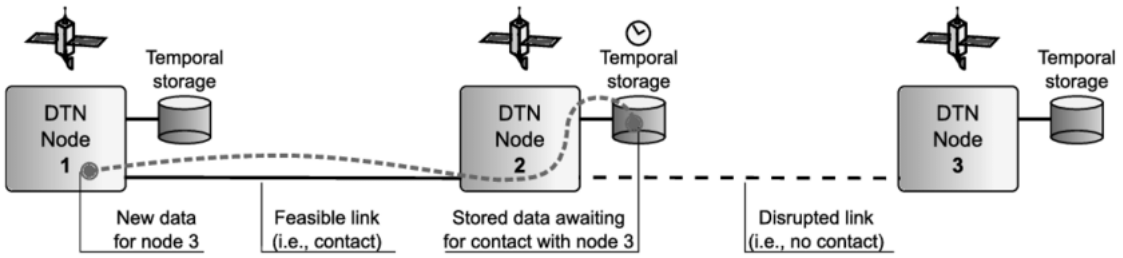

Figura 10. Mecanismo de almacenamiento y reenvío
Fuente: [26]

2.2.5 Control y gestión de flujo

En una DTN, el tamaño del buffer en los nodos y del ancho de banda son los recursos que se superan con mayor frecuencia y además suelen encontrarse limitados lo que resulta en un aumento de la congestión afectando al rendimiento de toda la red. En comparación con otras redes, el control de congestión en una DTN es relativamente difícil debido a dos características. Por un lado, es posible que con algún nodo en el futuro no se pueda establecer una conexión, por lo que no se puedan enviar datos acumulativos de anteriores nodos. Por otro lado, los paquetes con custodias aceptadas permanecen almacenados en el nodo hasta que caduquen o suceda un evento extremo como por ejemplo la destrucción del nodo. Según la RFC 4838 [9], el control de congestión se trata de un mecanismo que asegura que la tasa agregada a la que todas las fuentes de tráfico introducen datos en una red no exceda la tasa agregada máxima que la red puede entregar datos a los nodos de destino a lo largo del tiempo. Si el nodo DTN acepta demasiada información fuera del contrato de transferencia establecido, la red aparecerá congestionada. Si es así, el almacenamiento del nodo podría ser completamente consumido a largo plazo. En una DTN, el control de congestión es una de las partes más investigadas, ya que incluso después del despliegue exitoso de la red, los mensajes de alta prioridad deben ser supervisados para que no se pierdan, ya que puede afectar a la base de la DTN lo que conllevaría fallos en las aplicaciones. Un nodo recibe los paquetes si su búfer es capaz de retener todos los mensajes entrantes o, de lo contrario, pondrá en funcionamiento el mecanismo de control de congestión. Para evitar la transferencia de mensajes sin custodia, el mecanismo de control de congestión se puede dividir en dos tipos, proactivo y reactivo. El mecanismo proactivo normalmente evita la congestión enfocándose en el control de acceso al principio. Este método es práctico, ya que en muchos casos un área puede estar bajo el control de una sola

entidad. Si el mecanismo proactivo no está disponible, el reactivo se ejecutará a costa de un rendimiento deficiente. Algunos de estos mecanismos son:

- AFNER (Average Forwarding Number For Epidemic Routing/Numero de reenvío promedio): es un mecanismo reactivo utilizado en el enrutamiento epidémico, que supone que la información es transportada en paquetes (bundles) no fragmentados organizados con un número en orden ascendente de reenvío en el nodo emisor [27]. Mediante este mecanismo, el nodo de destino solo recibirá el paquete si el número del paquete es mayor que el promedio de los números de los paquetes que ya posee dentro de su buffer. De esta manera, cuando existe congestión, los paquetes con un numero de reenvío mayor o igual que el promedio de los paquetes almacenados en el búfer se eliminarán por el nodo de destino.
- N-Drop: es similar al mecanismo AFNER [28]. Se trata de un mecanismo reactivo que se puede utilizar en el enrutamiento epidémico. Si el nodo es capaz de recibir los mensajes entrantes los va almacenando hasta que se llena el búfer. En este momento, almacenara los paquetes borrando los que ya tiene almacenados, comparando el número de reenvío con el valor de umbral N que varía en función del tamaño del búfer. Si el búfer está lleno, empezará a borrar los paquetes con número de reenvío mayor que el valor de umbral y si todos los paquetes poseen número de reenvío menor que el valor de umbral N, se borrará el último paquete recibido.
- TBCC (Token Based Congestion Control/Control de congestión basado en tokens): a diferencia de AFNER y N-Drop, es un mecanismo de congestión proactivo e independiente del tipo de enrutamiento [29]. Solo el nodo que posea un token válido puede enviar mensajes en la red. Los tokens son distribuidos uniformemente y asignados posteriormente al azar a cada nodo con el objetivo de hacer coincidir la cantidad de datos que entran en la red con la capacidad total de la red. Esto significa que cualquier nodo o red DTN debe recibir solo la cantidad de datos que pueda entregar. Con TBCC, el nodo necesita un token para enviar información, que posteriormente es devuelto a la red cuando el mensaje sale de manera favorable o desfavorable. En comparación con otros mecanismos

de control de congestión, muestra un mejor desempeño porque evita la congestión antes de que haya ocurrido. Sin embargo, podría existir la situación en la que un nodo necesite enviar un mensaje de alta prioridad y por no tener tokens, el mensaje será descartado.

En una DTN, el control de flujo se refiere al control de la tasa de tráfico que discurre por un enlace entre dos nodos. Según la RFC 4838 [9], el control del flujo se trata del mecanismo que permite asegurar que la velocidad promedio a la que un nodo emisor envía datos, no es mayor que la velocidad promedio a la que el receptor está preparado para recibirlos. Las decisiones de control de flujo de DTN se deben tomar dentro de la propia capa del bundle en función de la información sobre los recursos disponibles dentro de los enlaces entre los nodos de la red. Si los recursos de almacenamiento están disponibles en otro lugar de la red, puede utilizarlos de alguna manera para el almacenamiento de paquetes. También puede descartar los paquetes que no hayan caducado, pero para los que no ha aceptado la custodia, sin embargo, un nodo debe evitar descartar los paquetes por los que ha aceptado la custodia y hacerlo solo como último recurso. Además de los mecanismos de capa bundle descritos anteriormente, un nodo DTN puede ser capaz de aprovechar el soporte de los protocolos de la capa inferior. Por ejemplo, un nodo DTN que recibe un paquete que utiliza TCP/IP puede ralentizar intencionalmente su velocidad de recepción al realizar operaciones de lectura con menos frecuencia para reducir la carga ofrecida. Esto es posible ya que TCP proporciona su propio control de flujo

2.2.6 Mecanismo de transferencia de custodia

En TCP, gracias a que los nodos de Internet se encuentran activos la mayor parte del tiempo, es posible proporcionar al usuario un servicio de transferencia fiable en el que los nodos de ambos extremos son los encargados de confirmar la correcta recepción de la información o solicitar un reenvío en caso de que los paquetes hayan sido entregados con errores o no entregados. Sin embargo, en una DTN debido a sus características, la conexión continua no se puede asegurar ya que la existencia de la unión extremo a extremo puede acabar en cualquier momento. Debido a la naturaleza intermitente de la conexión, DTN incorpora un mecanismo de transferencia de custodia que permite garantizar el reenvío fiable de datos hacia el próximo salto haciendo que algunos nodos

que utilizan el protocolo Bundle sean considerados como nodos custodios. En la red DTN, no es obligatorio que todos los nodos incorporen la transferencia de custodia por lo que no es un verdadero mecanismo salto a salto. La utilización de este mecanismo depende en muchas ocasiones de la capacidad del nodo en ese instante, ya que, si se encuentra congestionado o con poca energía, no podrá implementarlo. La transferencia de la custodia permite a la fuente, delegar la responsabilidad del reenvío y recuperar los recursos relacionados con la retransmisión en relativamente poco tiempo después de enviar el paquete. Al principio de la transmisión, el nodo custodio es la fuente del paquete y conforme avanza el paquete por los nodos, se va transfiriendo también la custodia. Para poder tener la custodia del paquete se deben cumplir una serie de requisitos por el nodo que recibe el paquete:

- Debe estar más cerca del destino final del paquete
- Tener disponibilidad para el almacenamiento durante un periodo de tiempo largo
- Tener la capacidad para posteriormente reenviar el paquete con el objetivo de entregarlo en el destino final.
- Poseer energía suficiente como para mantenerse activo durante un periodo de tiempo largo
- Trabajar de forma cooperativa con el resto de los nodos y aprovechar cada oportunidad que haya disponible para conseguir enviar el paquete al destino final.

El funcionamiento se muestra en la figura 11.

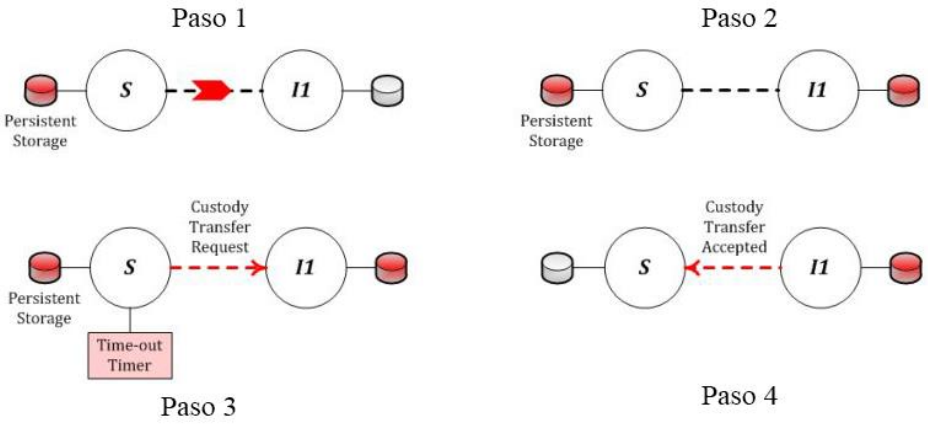

Figura 11. Funcionamiento del mecanismo de custodia
Fuente: [30]

En el primer paso, la fuente (S) encuentra al nodo I1 y realiza el reenvío del paquete que tenía en su almacenamiento. En el segundo paso, I1 guarda el paquete en su almacenamiento permanente. A continuación (paso 3), la fuente, una vez que ha considerado a I1 como un candidato valido para ser custodio, le envía una petición de transferencia de custodia y activa un temporizador. Si no se recibe una respuesta a la solicitud por parte de I1, el paquete se deberá reenviar seguido de una nueva solicitud. Si la solicitud es aceptada por I1 (paso 4), se debe responder a la fuente con un mensaje de confirmación y ya no se podrán descartar los paquetes de los que ha aceptado la custodia. Una vez que la confirmación es recibida por la fuente, borrará el paquete enviado de su almacenamiento y el proceso de transferencia de custodia habrá finalizado exitosamente. El proceso se realiza constantemente en todos los nodos que gracias a la capa Bundle tengan la capacidad para usarlo hasta el nodo final.

Cuando un nodo acepta la custodia de un paquete que tiene la opción de transferencia de custodia solicitada, la capa Bundle es la encargada de enviar la señal de aceptación de la custodia al actual EID que aparece en el bloque primario del Bundle. Posteriormente el EID de custodia actual es actualizado para que aparezca el EID de nodo que lo va a reenviar antes de enviarlo otra vez.

Una de las debilidades de este mecanismo implementado en la capa Bundle, consiste en que no posee ningún proceso de detección de errores o de rechazo de paquetes dañados. Otra de las debilidades se trata de que, en algunas ocasiones, el escenario DTN, solo existe comunicación unidireccional por lo que el mensaje de aceptación de custodia no podría ser recibido por el nodo de origen y por lo tanto el mecanismo de transferencia de custodia fallaría ya que caducaría la petición de transferencia de custodia.

2.2.7 Fragmentación

En una DTN, los paquetes pueden atravesar numerosas redes heterogéneas, lo que implica que cada una tendrá una limitación en cuanto al tamaño máximo de sus respectivas unidades de datos de protocolo (PDU). Por otro lado, en otras ocasiones, por ejemplo en comunicaciones satelitales, la capacidad del enlace es muy baja o el contacto es tan intermitente o de muy poca duración que no es posible reenviar el paquete completo. Es por ello que a veces los paquetes deben ser fragmentados en paquetes de

menor tamaño para poder llegar al destino final. La fragmentación de los paquetes se realiza para mejorar la eficiencia en la comunicación entre los nodos DTN siendo una garantía de utilización completa del contacto y evitando el reenvío de paquetes que ya habían sido parcialmente enviados.

El soporte de la fragmentación en el protocolo Bundle se hace de la misma forma que en IPv4. El campo *Fragment Offset* de la cabecera del paquete indica el desplazamiento y la longitud del fragmento en relación a la posición que ocupaba en el paquete original. La identificación que poseen los fragmentos de pertenencia al mismo paquete consiste en el EID del remitente y receptor y en la marca de tiempo (*Timestamp*) de creación. Los fragmentos de los paquetes solo son ensamblados de nuevo en el destino final.

Cualquier paquete en el que en los indicadores (*flags*) no se indique explícitamente que no debe ser fragmentado, es susceptible de ser fragmentado en cualquier momento. La fragmentación será gestionada del siguiente modo:

- La unión de la carga útil (*payload*) de todos los fragmentos siempre debe ser igual a la carga útil (*payload*) del paquete que se fragmentó.
- Los indicadores de procesamiento del paquete que se encuentran en la cabecera de cada fragmento deben indicar que el paquete es un fragmento.
- Si el bit "*Block must be replicated in every fragment*" que se encuentra en el indicador de control de la cabecera del paquete se establece a 1, entonces el bloque primario debe replicarse en cada fragmento.
- Si el bit "*Block must be replicated in every fragment*" que se encuentra en el indicador de control de la cabecera del paquete se establece en cero, el bloque primario se debe replicar en un solo fragmento.
- El orden relativo de todos los bloques que se encuentran en cada fragmento debe ser el mismo que existía en el paquete antes de la fragmentación.

Se definen dos tipos de fragmentación:

- Fragmentación proactiva: se utiliza cuando se conoce o se puede predecir la duración del contacto y la capacidad del enlace que se establece con el nodo de destino. Este tipo de fragmentación también puede ser utilizado para adaptar paquetes que estén orientados a mensajes de capa inferior. De esta manera, un

nodo DTN puede dividir el paquete varios bloques más pequeños y transmitir cada bloque como un paquete independiente.

- Fragmentación reactiva: se lleva a cabo la división de un paquete en fragmentos cuando los protocolos de la capa inferior a la capa Bundle indican que el paquete se transmitió parcialmente con éxito. En este caso, el nodo remitente elimina la parte transferida con éxito del paquete original y mantiene el fragmento no entregado para enviarlo en contacto posterior con el mismo nodo o con otro nodo. Por otro lado, la capa Bundle del nodo receptor modifica el paquete entrante para indicar que es un fragmento y así poder reenviarlo posteriormente. Es una fragmentación dinámica y por lo tanto más compleja de manejar que la fragmentación proactiva. No es obligatoria la implementación de este tipo de fragmentación en la capa Bundle de cada nodo ya que puede entrar en conflicto con el mecanismo de transferencia de custodia y con los mecanismos de seguridad de la DTN.

Por una parte, el mecanismo de transferencia de custodia ha desarrollado mejoras para poder soportar la fragmentación. Sin embargo, ya que los fragmentos de un mismo paquete pueden ser entregados a distintos nodos, dichos nodos tendrán la custodia de esos fragmentos del mismo paquete. Por lo tanto, es posible la existencia de más de un nodo custodio para los distintos fragmentos del mismo paquete.

La fragmentación representa un desafío respecto a la seguridad al requerir algunos paquetes el uso de firmas digitales. Cuando se recibe un paquete fragmentado que requiere de la firma digital, el código de autenticación fallara ya que la firma es verificada por cada nodo intermedio. En este caso, la verificación con firma digital y la fragmentación no son compatibles por lo que los paquetes que implementen este mecanismo deberán modificar el indicador de la cabecera del paquete para indicar que no puede ser fragmentado.

2.2.8 Capas de convergencia

El transporte con éxito del paquete desde el nodo de origen al nodo de destino depende de los protocolos subyacentes a la capa Bundle. La comunicación entre la capa Bundle y dichos protocolos de las capas subyacentes se llevan a cabo mediante las capas de

convergencia. Al utilizar el servicio del adaptador de la capa de convergencia, el protocolo Bundle puede realizar la entrega de los paquetes mientras usa el protocolo de transporte de paquetes adecuado para cada red. No todos los protocolos subyacentes proporcionan la misma funcionalidad por lo que dependiendo de cuál sea el escenario requerirá una adaptación distinta y por lo tanto también influirá en la complejidad de la capa de convergencia. Algunos de los ejemplos más importantes son:

- Licklider Transmission Protocol (LTP): este protocolo usado como capa de convergencia fue diseñado para ser utilizado en enlaces de radiofrecuencia del espacio exterior en conexiones punto a punto, de larga distancia o enlaces similares caracterizados por un retardo de transmisión extremadamente largo y posibilidad interrupciones frecuentes en la conexión. Al ser usado en conexiones punto a punto, no es necesario tener en cuenta problemas en el enrutamiento ni controlar la congestión. Todas sus características y funcionamiento se recogen en la RFC 5326 [11]. Este protocolo divide los datos en dos partes según el grado de fiabilidad en la entrega. Una parte roja que contiene información que debe ser entregada de forma fiable por lo que la recepción debe ser confirmada por el receptor y el nodo receptor debe mantenerse disponible hasta que lo confirme. La otra parte, llamada parte verde, contiene información cuya entrega no es estrictamente necesaria por lo que la recepción de los datos no se confirma y tras ser enviada se descarta en el nodo emisor. En este protocolo no se intercambian mensajes para realizar la negociación de la comunicación evitando de esta manera retrasos en la conexión.

  Teniendo como base este protocolo, se desarrolló el Licklider Transmision Protocol Transport (LTP-T) que es utilizado para escenarios con varios saltos con unos sistemas de fiabilidad en la entrega más complejos.
- Saratoga Protocol: es un protocolo ligero basado en el protocolo UDP utilizado para resolver los problemas en una conexión cuya tasa de transmisión es asimétrica, incluso pudiendo soportar una comunicación de datos unidireccional [31]. Saratoga hace un uso muy eficiente del tiempo de contacto rellenando completamente el enlace de paquetes, lo que garantiza la máxima transferencia

posible de datos. No implementa un mecanismo de congestión especifico ya que generalmente se usan enlaces de conexión privados.

- UDPCL: en este caso, la capa de convergencia está basada en el protocolo de datagramas de usuario (UDP) [32]. Está poco recomendado debido a los estrictos requisitos que se deben cumplir para hacer funcionar el protocolo Bundle sobre el protocolo UDP.
- TCPCL: esta capa de convergencia utiliza el protocolo de Control de Transmisión (TCP) como capa de transporte y sus características están recogidas en la RFC 7242 [33].

2.2.9 Seguridad

Desde el inicio del desarrollo del protocolo Bundle, se ha tenido muy en cuenta el diseño de la seguridad y para ello se ha creado el *Bundle Security Protocol* [13]. Dicho protocolo debe procurar que solo las entidades autorizadas de cada entorno puedan enviar paquetes, evitando el consumo desautorizado de recursos. Por otro lado, cada nodo debe garantizar la integridad de los datos que envía y recibe, así como de la confidencialidad de los paquetes cuando se encuentra circulando por la red ya que podrían llegar a nodos desautorizados. Por lo tanto, la seguridad en la capa Bundle está dirigida a la autenticidad, integridad y confidencialidad de los paquetes que circulan por la red, y para ello usa tres bloques de seguridad que se pueden usar de forma independiente o en conjunto dependiendo de las amenazas, requisitos y políticas de seguridad que existan en la red. Dichos bloques se pueden incluir en cada paquete y son los siguientes:

- Bloque de autentificación de Bundles (BAB): se genera cuando se envía o recibe cada paquete en el nodo y permite garantizar la autenticidad e integridad de los paquetes en cada nodo de la red. Cada nodo puede verificar la autenticidad del paquete gracias a este bloque de forma que se evitaran reenvíos por nodos no autorizados.
- Bloque de seguridad de la carga útil (PSB): en este caso, el bloque se genera en la fuente del paquete y se verifica la autenticidad cuando llega al destino final aunque existe la posibilidad de que también se cree y verifique en cada nodo de la ruta.

- Bloque de confidencialidad de la carga útil (PCB): la existencia de este bloque implica que algunas partes del paquete se han cifrado en el origen para proporcionar una mayor seguridad en el paquete. Este bloque indica el algoritmo y las ID de la clave que se han utilizado.

La seguridad del paquete no es invalidada si alguno de los nodos de la ruta no implementa el protocolo de seguridad de Bundle ya que su existencia es opcional, pero como se indica en la RFC 5050 [2] su uso está muy recomendado.

2.3 RESUMEN DEL CAPÍTULO

En este capítulo, inicialmente se ha ofrecido una visión teórica de las redes tolerantes al retardo y a las interrupciones así como de sus aplicaciones. Posteriormente se ha descrito el protocolo Bundle y su funcionamiento desgranando distintas características como son el mecanismo de transferencia de custodia, el sistema de almacenamiento y reenvío o las capas de convergencia que le permiten ofrecer un buen rendimiento en las DTN.

En el siguiente capítulo, se trasladarán los conocimientos teóricos presentados con anterioridad al apartado práctico desplegando el protocolo Bundle en un emulador de redes y obteniendo datos reales de su rendimiento en distintos escenarios.

CAPÍTULO 3.

EL PROTOCOLO BUNDLE EN UN EMULADOR DE REDES

En este capítulo se desarrollará el objetivo del trabajo que consiste en utilizar el protocolo Bundle dentro del emulador de redes *Common Open Research Emulator* (CORE) y comparar su rendimiento con el del protocolo TCP. Para ello, se desplegará el emulador de redes en una máquina virtual y se crearán diversos escenarios en los que probar el rendimiento de ambos protocolos. En cada escenario, se realizará un estudio de los valores obtenidos para comprobar en cuales resulta más adecuado el uso del protocolo Bundle respecto al protocolo TCP.

3.1 EL EMULADOR DE REDES CORE

El emulador de redes CORE es un programa de código abierto desarrollado inicialmente por Boeing y actualmente soportado por el Laboratorio de Investigación Naval de Estados Unidos [34]. CORE proporciona una interfaz gráfica al usuario (GUI) permitiendo crear redes fijas o móviles y generar tráfico entre ellas a tiempo real. Para ello, permite ejecutar una maquina Linux en cada nodo de la topología como si se tratase de un nodo real. También ofrece algunas funciones útiles para el usuario permitiendo conocer el tráfico que discurre por los enlaces o modificar cuales son las condiciones de los enlaces estableciendo distintos valores de retardo, ancho de banda…

Se ha utilizado un emulador de redes en lugar de un simulador debido a que permite utilizar implementaciones reales del protocolo y permite modificar variables que pueden no estar recogidas en un simulador. Además, el uso de un emulador facilita la interoperabilidad para usar otras implementaciones entre ellos, lo cual es complicado cuando se usa un simulador.

Actualmente, existen diversos emuladores de redes como OPNET, QualNet, NS-2 o *Opportunistic Networking Enviroment Simulator* (ONE) que es el más extendido, pero CORE ha sido el elegido para la realización del proyecto debido a los conocimientos previos que poseía el departamento de Ingeniería Telemática de la Universidad Carlos III de Madrid y por la facilidad que aporta CORE para poder reutilizar los escenarios y resultados obtenidos en estudios posteriores. Además, permite la implementación de

distintos protocolos en cada escenario, lo que en este caso es muy útil para establecer comparaciones entre el rendimiento del protocolo Bundle y TCP.

3.2 CONFIGURACIÓN DEL ENTORNO

Para la realización de las pruebas, se ha utilizado un ordenador portátil marca HP con arquitectura de 64 bits y Windows 10. Posee un procesador Intel Core i7 6700-HQ y 16 GB de memoria RAM. El emulador de redes CORE no funciona sobre Windows por lo que se ha instalado el software de virtualización Virtual Box en su versión 5.2.18 en el que se ha creado una máquina virtual con Linux Ubuntu (64-bit) ejecutándose sobre ella.

La versión de CORE utilizada es la más reciente hasta la fecha y es la versión 4.8 obtenida de su página web de descargas. Por otro lado, para la obtención de capturas de los paquetes que circulan por las redes creadas se ha utilizado el analizador de protocolos Wireshark [35] en la versión 2.6.6. Además, se utilizará dicho programa para la obtención de estadísticas temporales sobre los protocolos TCP y Bundle observando el intercambio de paquetes generado.

3.3 IMPLEMENTACION IBR-DTN Y ALTERNATIVAS

Para la utilización del protocolo Bundle en el emulador CORE, se ha utilizado la implementación IBR-DTN. Esta implementación, basa su funcionamiento en la RFC 5050 [2] del protocolo Bundle y la RFC 6257 [37] del protocolo de seguridad de Bundle y fue desarrollada por la *Technical University of Braunschweig* (Alemania) en 2012. Permite su ejecución en Debian, Ubuntu, Android, MacOS X, Raspberry Pi y BeagleBone.

IBR-DTN ha sido la implementación elegida debido a que es muy ligera al hacer un menor uso de CPU y de consumo de memoria [36] por lo que es muy adecuada para ejecutar en los nodos del emulador CORE. Por otro lado, es una implementación portable basada en módulos que le permiten adaptarse a la características del entorno en el que se quiera probar. Al instalarse esta implementación se incluyen algunos módulos por defecto entre los que están las capas de convergencia TCPCL (elegida en este trabajo), UDPCL, HTTPCL y LowPANCL. Otro de los módulos de esta implementación está dedicado al tipo de almacenamiento en los nodos permitiendo guardar los archivos en memoria de forma no persistente (configurado por defecto), de forma persistente si se le indica un directorio y en formato de base de datos SQLite, aún en fase de pruebas.

Para seleccionar el tipo de enrutamiento también se incluye un módulo que permite al usuario elegir entre el enrutamiento PRoPHET, por inundación, epidémico o a través de rutas estáticas que deben ser configuradas.

IBR-DTN incorpora varias aplicaciones que nos permiten realizar distintos tipos de pruebas introduciendo los comandos adecuados. Sus aplicaciones más importantes son:

- dtnping: permite probar la conexión entre dos nodos mediante paquetes Bundle.
- dtnrecv/dtnsend: con esta aplicación mediante el comando dtnrecv se crea un directorio de recepción de archivos y mediante el comando dtnsend se indica manualmente cual es el nodo de destino y el archivo que se quiere enviar. Esa aplicación ha sido la utilizada para realizar las pruebas.
- dtninbox/dtnoutbox: es una aplicación similar a la anterior que permite crear una carpeta de entrada y de salida en distintos nodos, de manera que cuando se introduce un archivo en la carpeta de salida, posteriormente se envía a la carpeta de entrada de forma automática.
- dtntunnel: permite la creación de un túnel DTN por el que pueden ser enrutados los paquetes.
- dtntracepath: mediante este comando, se puede comprobar cuál es la ruta hacia el nodo indicado .
- dtnstream: permite recibir tráfico en directo desde Internet.

En el Anexo B, se encuentran los pasos que se han llevado a cabo para configurar la implementación IBR-DTN y su posterior uso en CORE.

Aunque la implementación IBR-DTN fue la elegida para su uso en este trabajo, previamente se estudió y valoraron otras opciones con un uso muy extendido como son:

- DTN2: es la implementación de referencia para una DTN creada por el DTNRG en lenguaje C++. Soporta varias capas de convergencia y protocolos de enrutamiento, pero no ha recibido actualizaciones desde 2012.
- ION-DTN: desarrollada por el *Jet Propulsion Laboratory* de la NASA en lenguaje C, se centra en las comunicaciones espaciales y es actualizada frecuentemente siendo su última versión de diciembre de 2018.

Se descartó la utilización de la implementación DTN2 ya que a pesar de ser la utilizada como referencia para las DTN, no se encuentra actualizada desde hace 7 años. Por otro lado, se planteó la utilización de la implementación ION-DTN pero debido a IBR-DTN es más ligera y cumplía con las necesidades del trabajo, finalmente fue la elegida. Además, también se ha valorado en la elección de IBR-DTN que actualmente posee soporte por la *Technical University of Braunschweig* habiendo sido actualizada por última vez en agosto de 2018 para mejorar la seguridad del protocolo.

3.4 DESARROLLO Y RESULTADO DE LAS PRUEBAS

Para la realización de las pruebas ejecutaremos el emulador CORE en la máquina virtual que hemos creado, utilizando los comandos:

```
sudo core-daemon start

core-gui
```

Al introducir estos comandos, se lanzará la interfaz gráfica de la figura 12 en la que podremos crear distintos tipos de redes y variar diversas características en los enlaces (ancho de banda, retrasos, porcentaje de pérdida de paquetes) para probar el rendimiento del protocolo Bundle y compararlo con el de TCP.

Para las pruebas del protocolo Bundle, se ha utilizado TCPCL como capa de convergencia y se ha modificado el archivo de configuración de la implementación IBR-DTN que se lanza en cada nodo. En este archivo, que se explica con detalle en el Anexo B, se han modificado algunos valores importantes como son el tipo de enrutamiento y el tamaño de almacenamiento máximo permanente en cada nodo. Para realizar el envío de los archivos se ejecutarán los comandos dtnrecv en el nodo receptor y dtnsend en el nodo emisor.

Para las pruebas del protocolo TCP, se utilizará como aplicación FTP (protocolo de transferencia de archivos) en el que uno de los nodos será el servidor y el nodo final será el cliente.

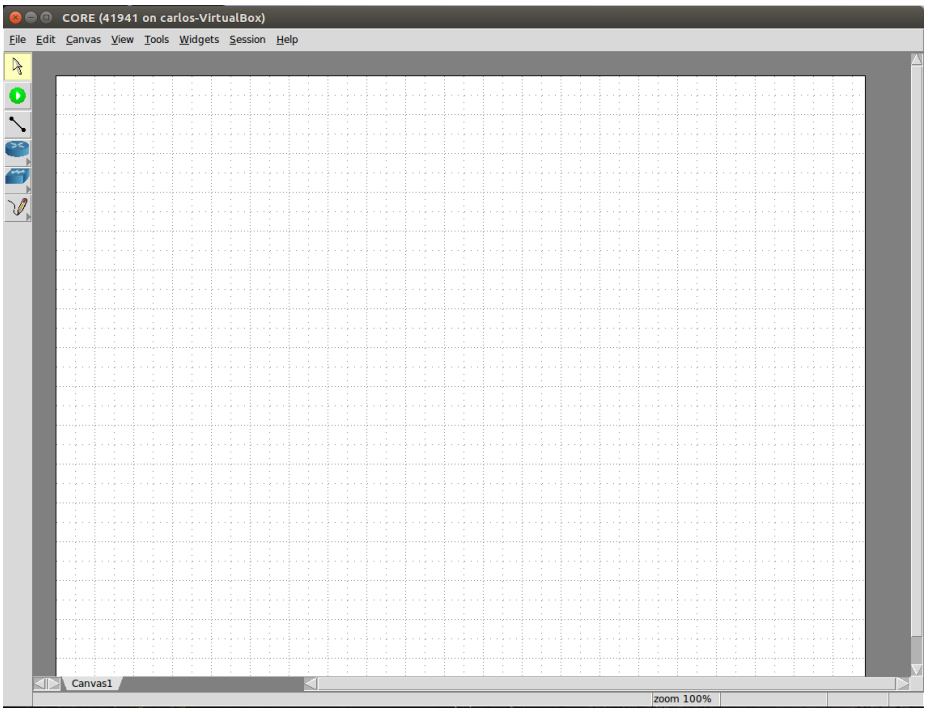

Figura 12. Interfaz gráfica del emulador CORE

En la realización de cada prueba, se harán 20 ejecuciones, reiniciando la simulación en todos los casos para evitar la posible influencia de anteriores ejecuciones. Como medida del rendimiento de ambos protocolos se ha recogido cuál es el valor de la tasa de transferencia (*goodput*) en cada caso calculada como el tamaño de un fichero transmitido dividido por el tiempo desde que el nodo de origen envía el primer paquete del fichero hasta que llega el último paquete al nodo de destino final, sin tener en cuenta el tiempo de establecimiento de la conexión TCP (que solo afectaría a FTP ya que Bundle no necesita abrir la conexión cuando va a enviar el fichero, ya que lo hace solo una vez al lanzar el demonio). De esta forma, se puede realizar una comparación justa entre ambos protocolos ya que se calcula el tiempo exacto que tarda en enviarse un archivo completo desde el nodo de origen hacia el destino en relación con el tamaño del archivo. Para la toma de medidas sobre el tiempo se utilizará el programa Wireshark. En todas las pruebas se realizará el envío de archivos de tamaños distintos y se tomará un intervalo de confianza del 90%.

3.4.1 Escenario simple

En este primer escenario con la estructura de la figura 13, se tomarán los valores de referencia para ambos protocolos.

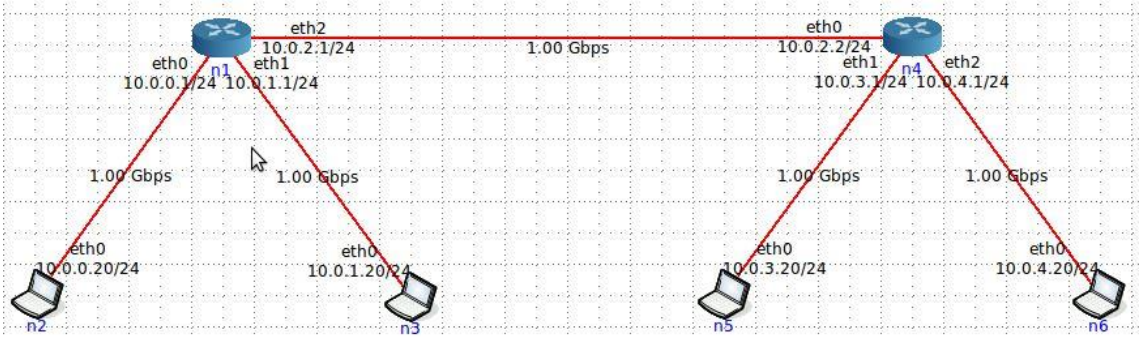

Figura 13. Escenario simple

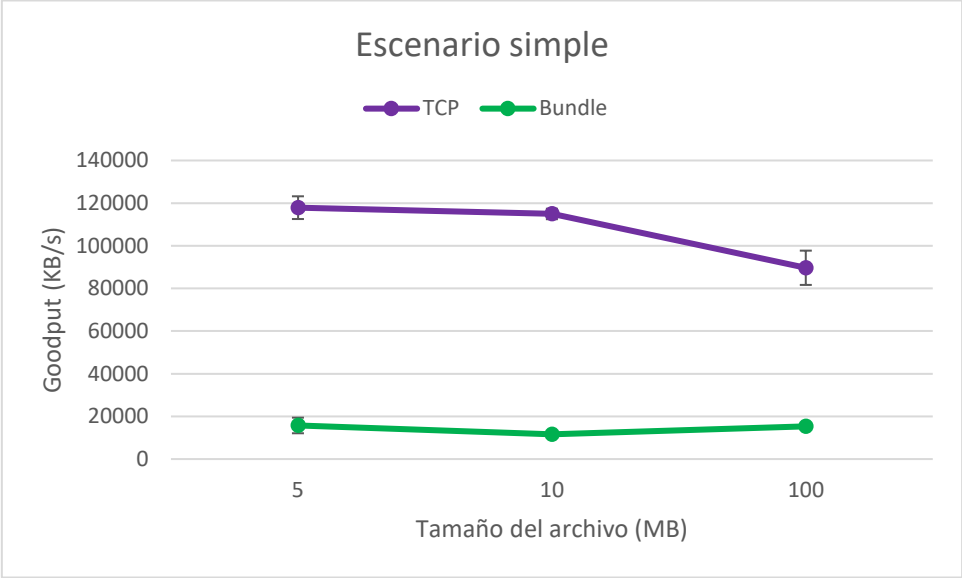

Figura 14. Valores de goodput del escenario simple

	Goodput (kB/s)		Desviación típica (σ)		Intervalo de confianza (90%)	
	Bundle	**TCP**	**Bundle**	**TCP**	**Bundle**	**TCP**
5 MB	15752,2879	117863,286	4984,1716	8570,55939	3666,3612	5328,2845
10 MB	11598,3845	114919,429	1134,5552	3949,45819	834,579837	2455,36328
100 MB	15288,051	89681,0429	1331,6146	12922,1935	979,536938	8033,67905

Tabla 3. Valores del escenario simple

En este escenario que se toma como referencia ya que no tienen ningún tipo de retardo ni pérdida, se puede comprobar como el goodput para TCP se acerca mucho al límite del ancho de banda de 1 Gbps teniendo valores de hasta 920 Mbit/s de goodput. Sin embargo, se puede comprobar que el rendimiento del protocolo Bundle en un escenario con condiciones de red normales es muy inferior, por lo que no está recomendado su uso en este tipo de entornos.

3.4.2 Escenario con retardo

En este primer caso, se creará una red como en la figura 15 en la que se disponen dos routers interconectados entre ellos y dos PCs conectados a cada uno. En este escenario, se introducirán distintos valores de retardo (500 milisegundos, 1 segundo y 4 segundos) entre los router n1 y n4.

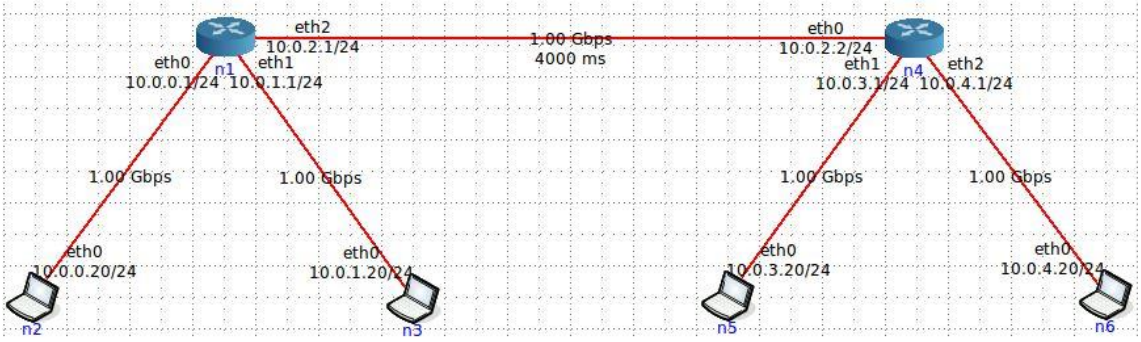

Figura 15. Escenario con retardo

Después de realizar las pruebas con ambos protocolos, se obtienen los siguientes valores:

- Retardo de 500 milisegundos

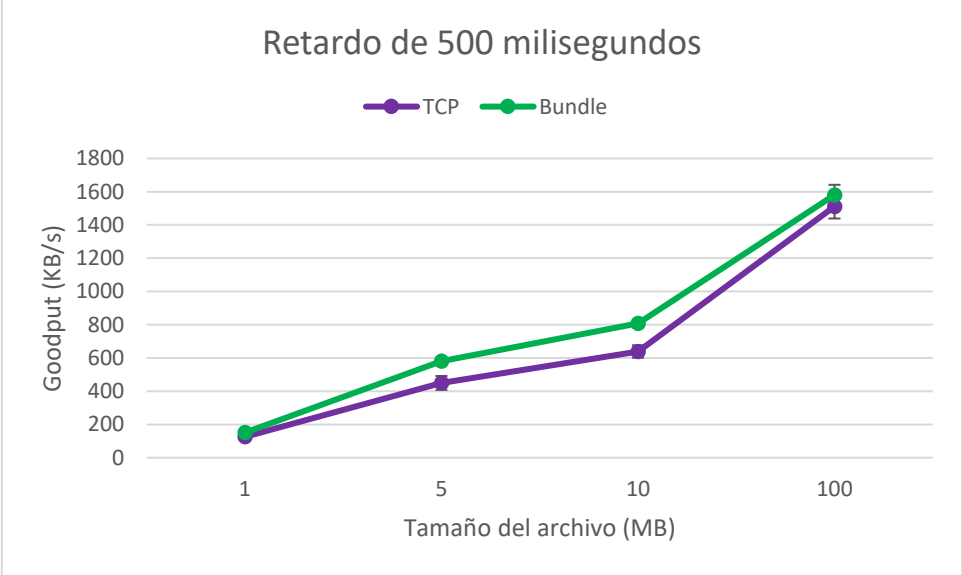

Figura 16. Valores de goodput con retardo de 500 milisegundos

	Goodput (kB/s)		Desviación típica (σ)		Intervalo de confianza (90%)	
	Bundle	**TCP**	**Bundle**	**TCP**	**Bundle**	**TCP**
1 MB	150,92317	125,508334	3,96800217	14,1458292	2,91886599	10,4056847
5 MB	581,58961	449,941276	4,05664277	56,5190119	2,98407009	41,5754363
10 MB	807,00416	638,964828	31,4331522	50,7294029	23,1222552	37,3165947
100 MB	1580,5048	1561,33425	82,870588	105,174637	60,9596798	77,3665584

Tabla 4. Valores del escenario con retardo 500 milisegundos

- Retardo de 1 segundo

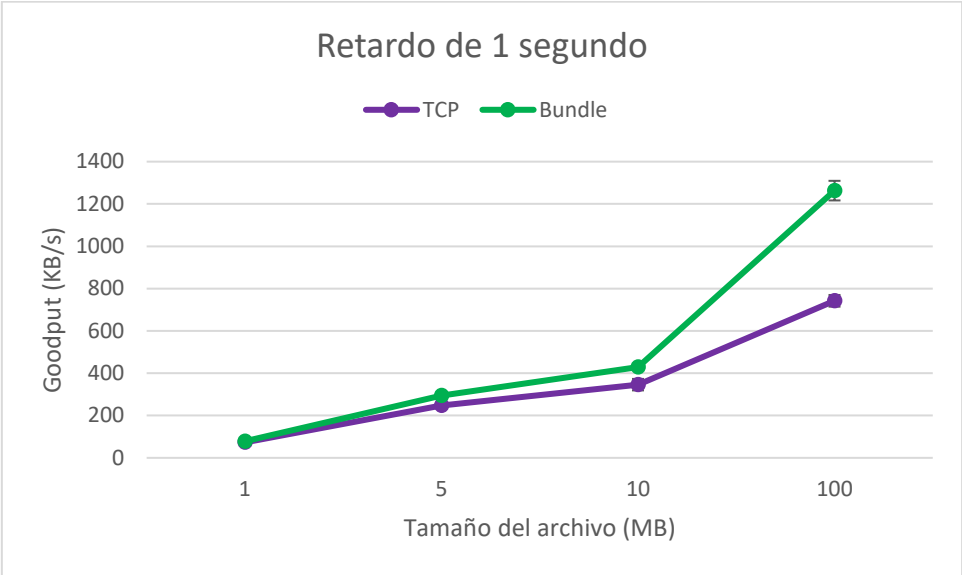

Figura 17. Valores de goodput con retardo de 1 segundo

	Goodput (kB/s)		Desviación típica (σ)		Intervalo de confianza (90%)	
	Bundle	**TCP**	**Bundle**	**TCP**	**Bundle**	**TCP**
1 MB	78,367201	72,9763859	0,47754484	0,15524959	0,35128242	0,11420174
5 MB	294,79845	247,562782	3,69963075	12,8122102	2,72145172	9,42467343
10 MB	428,45070	345,864793	2,58908689	36,9704112	1,90453466	27,1954679
100 MB	1262,9247	742,217194	63,0072445	37,4370811	46,348186	27,5387507

Tabla 5. Valores del escenario con retardo 1 segundo

- Retardo de 4 segundos

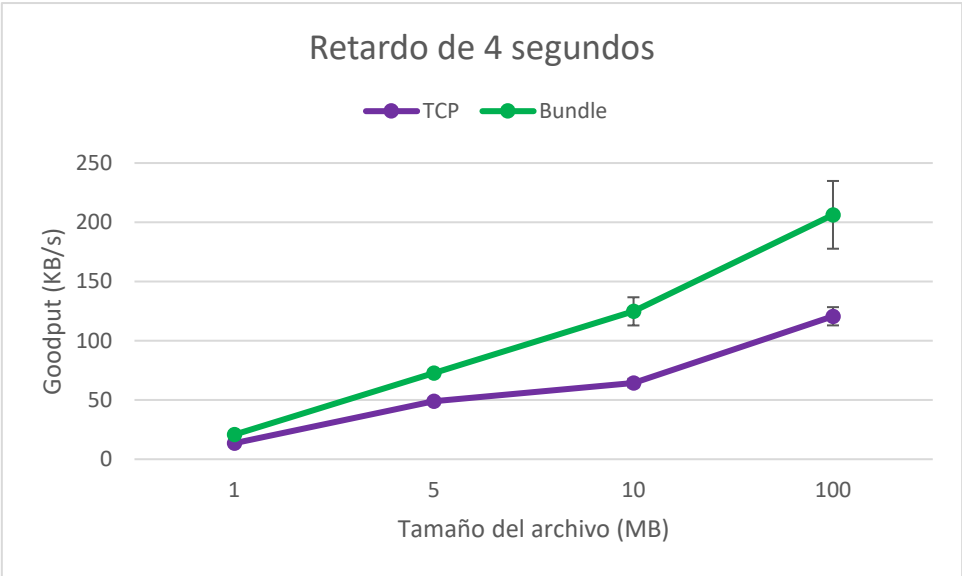

Figura 18. Valores de goodput con retardo de 4 segundos

	Goodput (kB/s)		Desviación típica (σ)		Intervalo de confianza (90%)	
	Bundle	**TCP**	**Bundle**	**TCP**	**Bundle**	**TCP**
1 MB	20,71280	13,49997	0,93998188	0,89623508	0,69145152	0,65927134
5 MB	72,683113	49,05006	3,65389197	2,46586597	2,68780624	1,81389324
10 MB	124,80885	64,2876084	16,1604074	4,90586327	11,8876103	3,60875746
100 MB	206,26762	120,665368	38,8745093	6,62360421	28,5961242	7,70382897

Tabla 6. Valores del escenario con retardo 4 segundos

En el escenario de retardo en un enlace, se pretende comprobar la eficiencia del protocolo Bundle simulando los largos retardos que se pueden producir a causa de las largas distancias que puede cubrir una DTN. En este caso, el valor de goodput para TCP es menor que para Bundle debido a los mecanismos de *Slow Start* y de control de congestión. El mecanismo de Slow Start, no resulta eficiente en este tipo de escenarios ya que el nodo de origen al no conocer las características de la red, empieza la comunicación con una tasa de envío muy pequeña. Por otro lado, el mecanismo de congestión de TCP funciona de tal manera que la ventana de congestión se aumenta progresivamente al recibir el ACK de confirmación de recepción. Debido a que en este escenario existen retardos, el tiempo que tarda en llegar el ACK será muy grande y, por lo tanto, la ventana de congestión crecerá muy lentamente afectando negativamente al valor de goodput.

Por el contrario, con el protocolo Bundle, la conexión se mantiene abierta permanentemente a partir de la ejecución del demonio en dos nodos que se encuentren unidos. De esta forma que cada nodo, conoce cuales son las características de la red y posibilita que desde el principio se pueda realizar la comunicación con una tasa de envío más alta. La existencia de una conexión permanente entre dos nodos es debida a que, durante el establecimiento de la conexión, en el paquete Contact Header intercambiado entre ambos nodos, existe el campo Keep Alive que se observa en la figura 19 que indica la frecuencia (en este caso 60 segundos, por defecto) con la que se intercambiaran los paquetes KEEPALIVE que mantienen la conexión abierta.

```
▶ Frame 309: 83 bytes on wire (664 bits), 83 bytes captured (664 bits) on interface 0
▶ Ethernet II, Src: 00:00:00_aa:00:00 (00:00:00:aa:00:00), Dst: 00:00:00_aa:00:01 (00:00:00:aa:00:01)
▶ Internet Protocol Version 4, Src: 10.0.0.20, Dst: 10.0.0.1
▶ Transmission Control Protocol, Src Port: 50364, Dst Port: 4556, Seq: 1, Ack: 18, Len: 17
▼ DTN TCP Convergence Layer Protocol
  ▼ Contact Header
      Magic: dtn!
      Version: 3
    ▼ Flags: 0x07
        .... ...1 = Bundle Acks Requested: True
        .... ..1. = Reactive Fragmentation Enabled: True
        .... .1.. = Support Negative Acknowledgements: True
      Keep Alive: 60
      Local EID: dtn://n2
  Local EID Length: 8
```

Figura 19. Campo Keep Alive del paquete Contact Header

En este mismo escenario también se han tomado medidas de latencia, es decir del periodo de tiempo que tarda el primer paquete desde que sale del nodo de origen hasta que llega al nodo del destino y por otro lado también se ha medido el periodo de tiempo que discurre desde que llega el primer paquete al nodo de destino hasta que llega el último (goodput en destino). Estas medidas en el escenario con retardo de 1 segundo se pueden observar en las figuras 20 y 21, y las medidas en el escenario con 4 segundos de retardo se pueden observar en las figuras 22 y 23.

- Retardo de 1 segundo

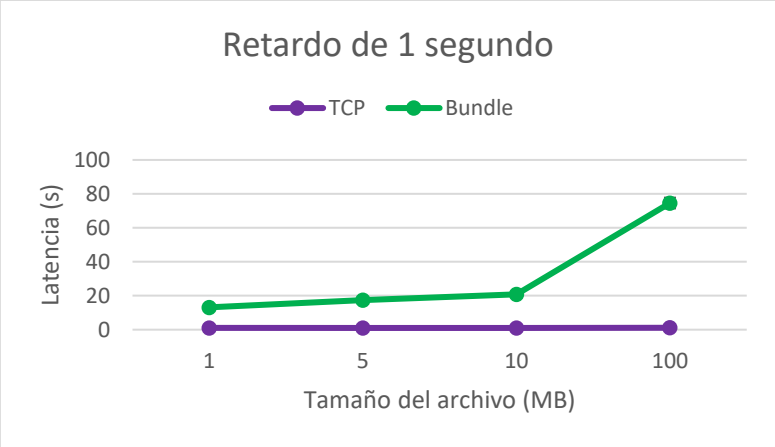

Figura 20. Valores de latencia con retardo 1 segundo

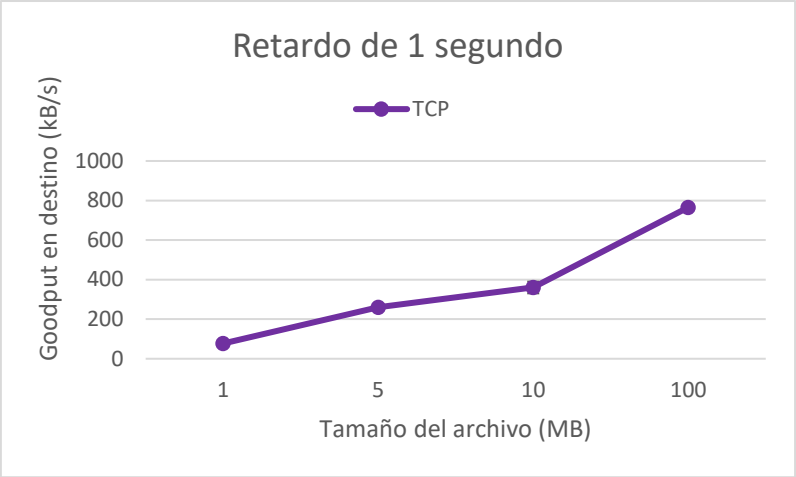

Figura 21. Goodput desde que llega el primer paquete hasta que llega el último al nodo de destino con retardo 1 segundo

- Retardo de 4 segundos

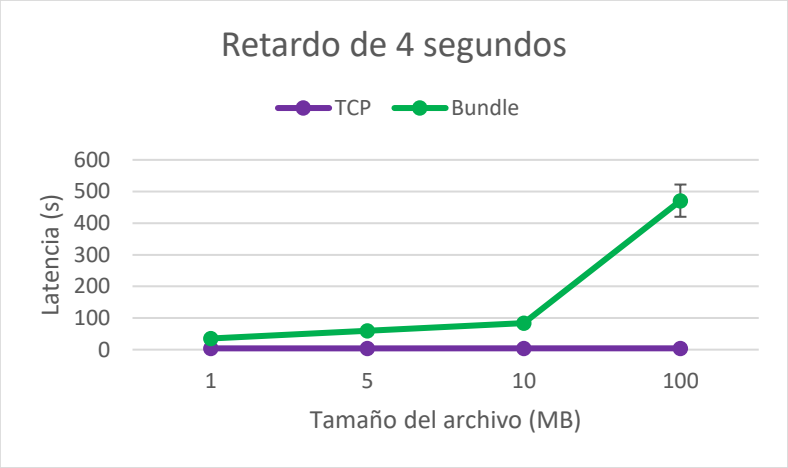

Figura 22. Valores de latencia con retardo 4 segundos.

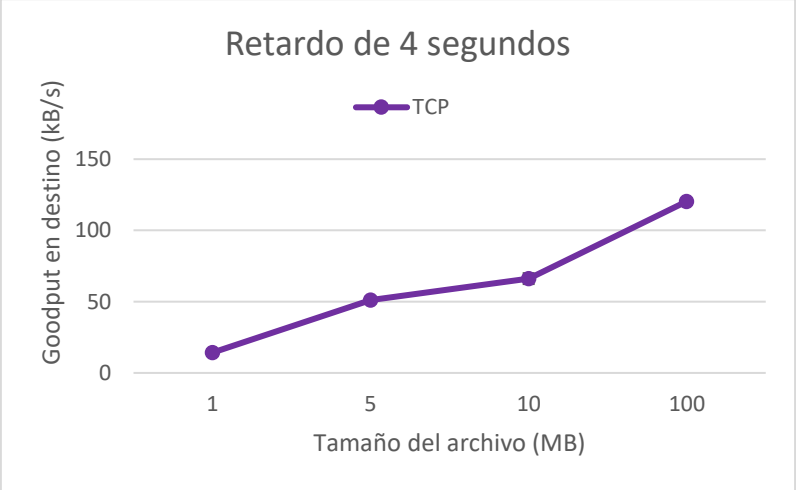

Figura 23. Goodput desde que llega el primer paquete hasta que llega el último al nodo de destino con retardo 4 segundos.

Gracias a la toma de estas medidas, se pueden comprobar grandes diferencias entre ambos protocolos. En la medida de la latencia, se puede comprobar como el protocolo TCP puede hacer llegar el primer paquete del archivo en un periodo de tiempo prácticamente igual al del retardo debido a que los paquetes no van pegados al anterior como ocurre en Bundle y viajan de forma independiente. Además en el periodo de latencia de Bundle también influye el tiempo de almacenamiento necesario en cada nodo por el que transitan los paquetes. Por el contrario, dado que los paquetes en el protocolo Bundle viajan de nodo en nodo, el tiempo que discurre desde que llega el primer paquete al nodo de destino hasta que llega el ultimo, es muy pequeño, ya que llegan casi prácticamente todos a la vez al destino final y por ello el goodput en Bundle es muy alto (por lo que no se ha mostrado en las gráficas). Este hecho es debido a que en el caso de Bundle el goodput calculado es prácticamente el del ultimo enlace mientras que el de TCP es el resultado de atravesar los 3 enlaces.

3.4.3 Escenario con pérdidas

En este caso, como se observa en la figura 24 se trata de la misma arquitectura de red que en el escenario anterior. Sin embargo en esta ocasión, el enlace que une los dos routers posee distintos porcentajes de pérdida de paquetes (10% y 20%).

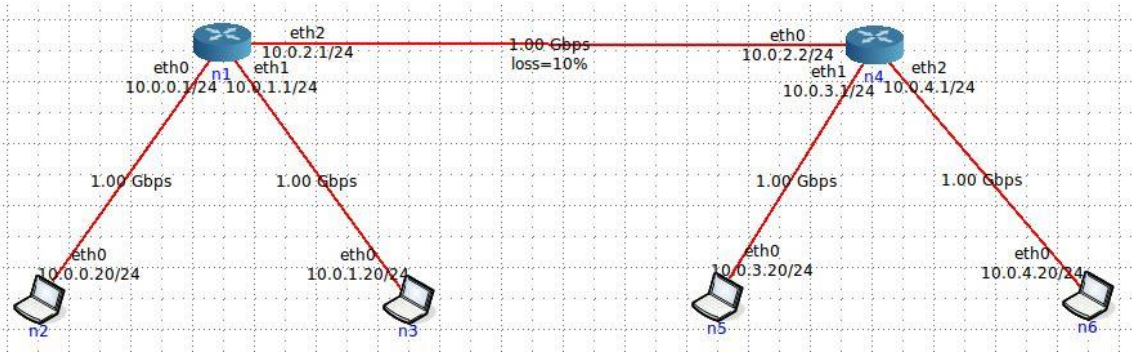

Figura 24. Escenario con pérdidas

Se realizaron pruebas con ambos protocolos y se obtuvieron los siguientes resultados:

- Pérdidas del 10%

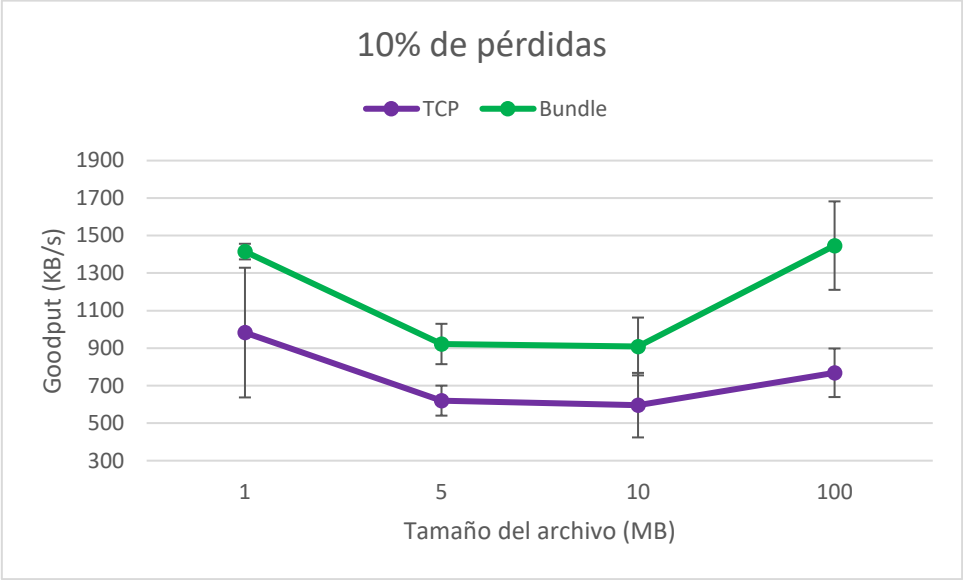

Figura 25. Valores de goodput con 10% de pérdidas

	Goodput (kB/s)		Desviación típica (σ)		Intervalo de confianza (90%)	
	Bundle	**TCP**	**Bundle**	**TCP**	**Bundle**	**TCP**
1 MB	1414,7900	982,925217	56,610056	470,124457	41,6424084	345,823976
5 MB	922,14054	620,153302	146,238932	108,755043	107,57349	80,0003077
10 MB	908,78560	595,949469	209,613399	233,862937	154,191806	172,029788
100 MB	1446,6826	768,650474	320,583207	176,026106	235,821297	129,484962

Tabla 7. Valores del escenario con 10% de pérdidas

- Pérdidas del 20%

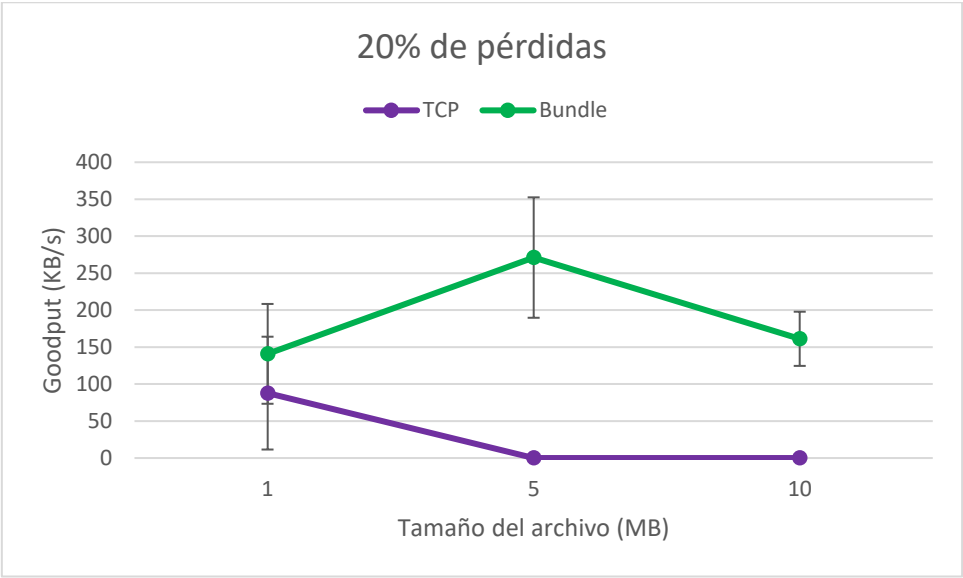

Figura 26. Valores de goodput con 20% de pérdidas

	Goodput (kB/s)		Desviación típica (σ)		Intervalo de confianza (90%)	
	Bundle	**TCP**	**Bundle**	**TCP**	**Bundle**	**TCP**
1 MB	140,874043	87,7927487	91,7403465	103,715962	67,4842819	76,2935557
5 MB	271,094716	---------	110,682979	---------	81,4184997	---------
10 MB	161,23782	---------	49,8088288	---------	36,6394195	---------

Tabla 8. Valores del escenario con 20% de pérdidas

- Pérdidas del 25%

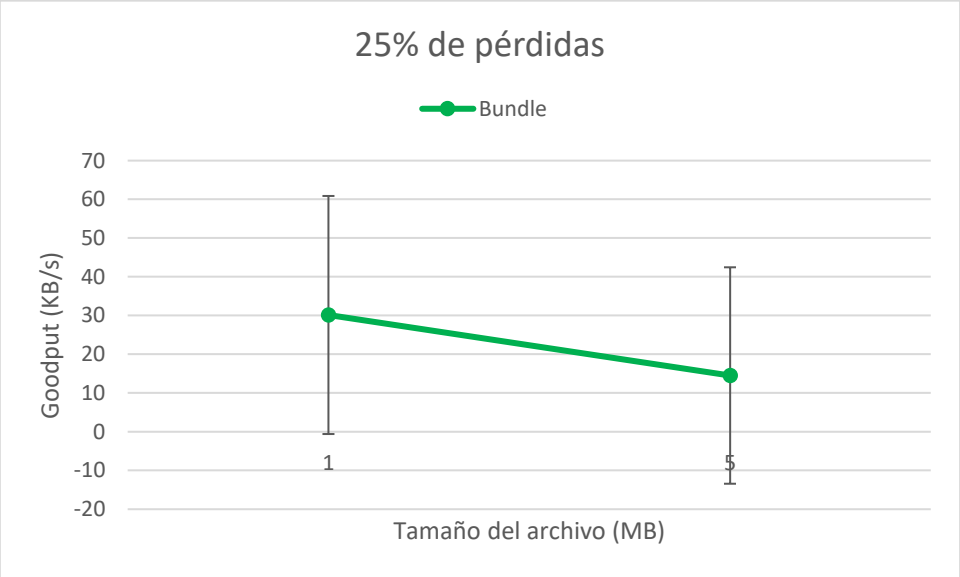

Figura 27. Valores de goodput con 25% de pérdidas

	Goodput (kB/s)		Desviación típica (σ)		Intervalo de confianza (90%)	
	Bundle	**TCP**	**Bundle**	**TCP**	**Bundle**	**TCP**
1 MB	30,144050	---------	41,7760849	---------	30,7305258	---------
5 MB	14,521531	---------	37,9828577	---------	27,9402245	---------
10 MB	---------	---------	---------	---------	---------	---------

Tabla 9. Valores del escenario con 25% de pérdidas

En los escenarios con distinto porcentaje de pérdidas, se puede observar como a medida que las pérdidas aumentan, los rendimientos de Bundle y TCP disminuyen considerablemente. En ambos casos, los valores de goodput tienen una gran variación debido a la aleatoriedad del paquete que se pierde.

Para poder observar mejor las diferencias de rendimiento en este escenario se han realizado las figuras 28 y 29 con la ayuda del programa Wireshark, en las que se puede observar la tasa de transferencia (throughput) de un archivo de 10 MB entre los dos routers del escenario con 10% de pérdidas.

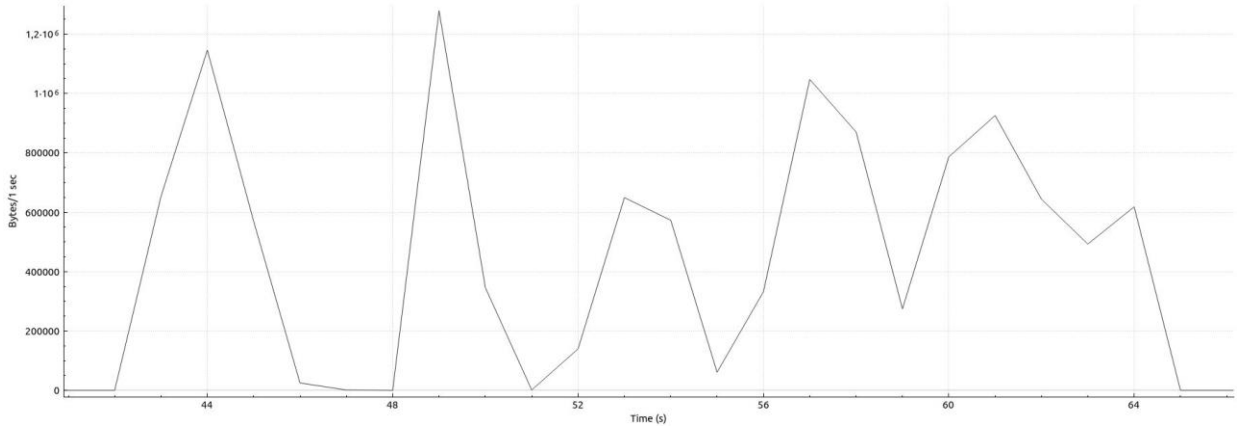

Figura 28. Gráfica de throughput con TCP en escenario de pérdidas

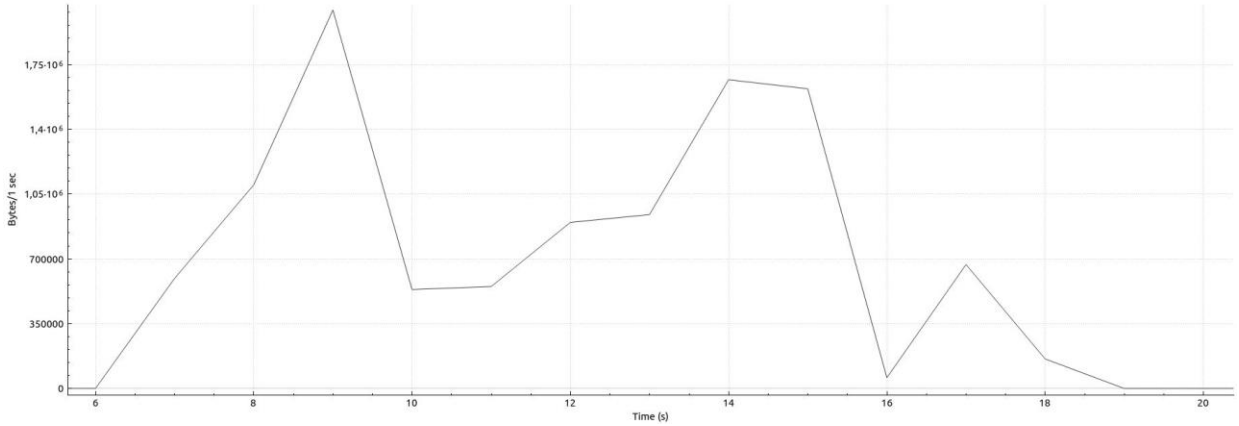

Figura 29. Grafica de throughput con Bundle en escenario de pérdidas

En la figura 28 referida al protocolo TCP se puede observar cómo se dan 3 situaciones en las que la tasa de transferencia se reduce prácticamente a cero. Sin embargo, en el caso del protocolo Bundle, solo hay un momento, en el segundo 16, en el que la tasa de transferencia es cero, y aunque también hay bajadas de throughput, estas son menos acusadas que con TCP.

Esta diferencia radica en que en el escenario en el que se usa TCP, se producen retransmisiones del mismo segmento debido a que se detectan pérdidas y por lo tanto este hecho, produce que el temporizador de retransmisión crezca de forma consecuente y se realicen paradas en los envíos. Debido a que estamos usando la capa de convergencia TCPCL, esperaríamos un efecto similar para el protocolo Bundle en el enlace entre los dos routers (el enlace con pérdidas), ya que está basada en TCP. Este efecto requiere más análisis y, por ello, es posible abrir una vía de investigación en el futuro para comprobar cuál es la diferencia de funcionamiento en este escenario.

Por otro lado, con TCP en algunos casos (con pérdidas superiores al 25%) se produce una pérdida completa de la comunicación mientras que con el protocolo Bundle sigue siendo posible a pesar de tener un goodput muy reducido. Este hecho es debido a que con el protocolo Bundle aunque se pierda la conexión debido a las pérdidas, la capa TCPCL se encarga de abrir conexiones cada cierto tiempo (mediante los paquetes KEEPALIVE) de manera que cuando consiga volver a conectar con el nodo siguiente, podrá continuar la transmisión desde el punto en el que se había quedado (ya que TCPCL lleva cuenta de los bundles transmitidos con éxito y los que no). En el caso de TCP, aunque pudiese abrir la conexión (algo que FTP no hace de forma automática) debería empezar a transmitir desde el principio.

Aunque en este escenario no tiene efecto ya que solo hay un enlace con pérdidas, una ventaja adicional de Bundle es que el mecanismo de transferencia de custodia implementado en el protocolo Bundle, permitiría la recuperación de las pérdidas en el otro lado del enlace, en vez de extremo a extremo como ocurre con TCP ya que el protocolo Bundle gestiona cada enlace independientemente.

3.4.4 Escenario con ancho de banda asimétrico

En este caso mostrado en la figura 30, los elementos de red utilizados serán de nuevo dos routers y cuatro PCs. En este escenario, se realizarán las pruebas observando el rendimiento de ambos protocolos con valores de ancho de banda asimétricos en el enlace que une los dos routers, es decir, no tendrá el mismo valor el ancho de banda de subida que de bajada. Se realizaron pruebas de rendimiento con varios valores (10 Mbps para el envío de datos / 100 Mbps para recibir ACKs y 5 Mbps para el envío de datos / 50 Mbps para recibir ACKs) y se obtuvieron los siguientes resultados:

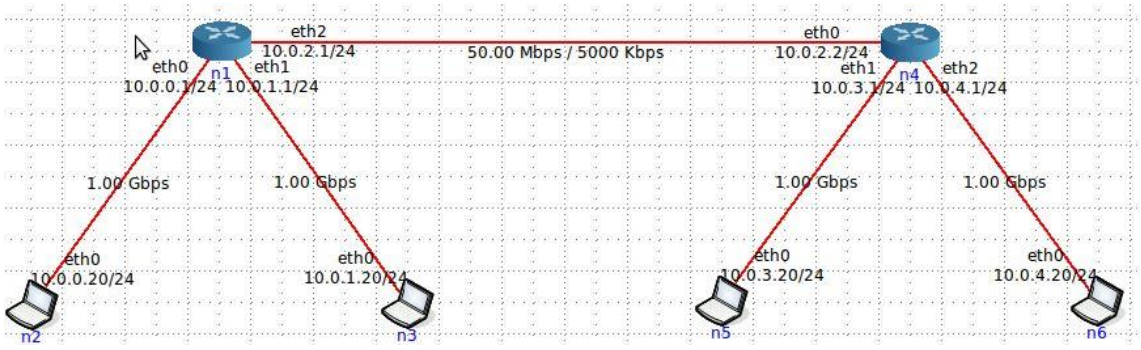

Figura 30. Escenario con ancho de banda asimétrico

- Ancho de banda 10 Mbps / 100 Mbps

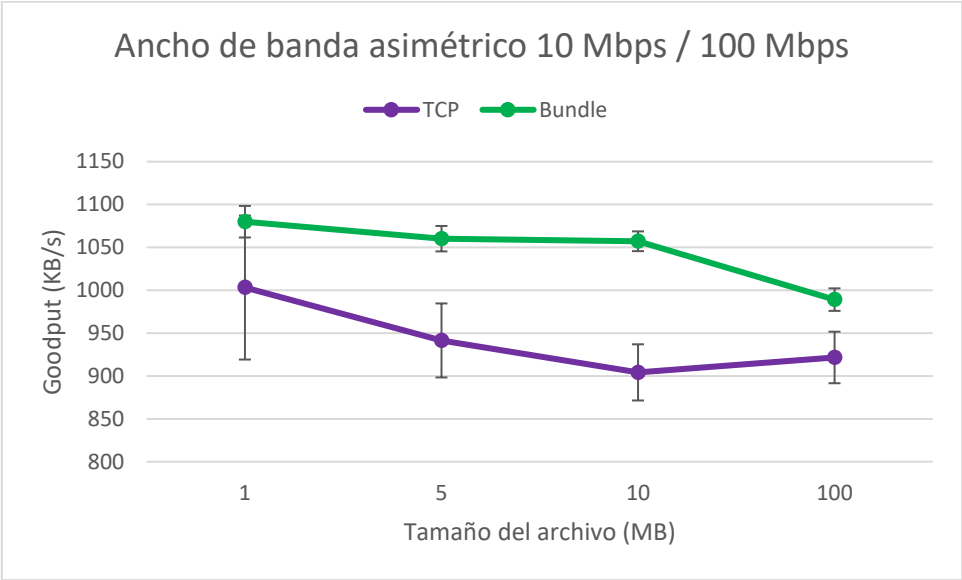

Figura 31. Valores de goodput con ancho de banda 10 Mbps / 100 Mbps

	Goodput (kB/s)		Desviación típica (σ)		Intervalo de confianza (90%)	
	Bundle	**TCP**	**Bundle**	**TCP**	**Bundle**	**TCP**
1 MB	1079,922	1003,15432	35,3922747	161,493428	18,4092346	84,0005461
5 MB	1060,145	941,437995	28,4872919	83,0082729	14,8176189	43,1766193
10 MB	1057,110	904,165025	22,1688927	62,9983329	11,5311138	32,7684813
100 MB	989,0558	921,586205	25,2357991	57,7847227	13,1263602	30,0566303

Tabla 10. Valores del escenario con ancho de banda 100 Mbps / 10 Mbps

- Ancho de banda 5 Mbps / 50 Mbps

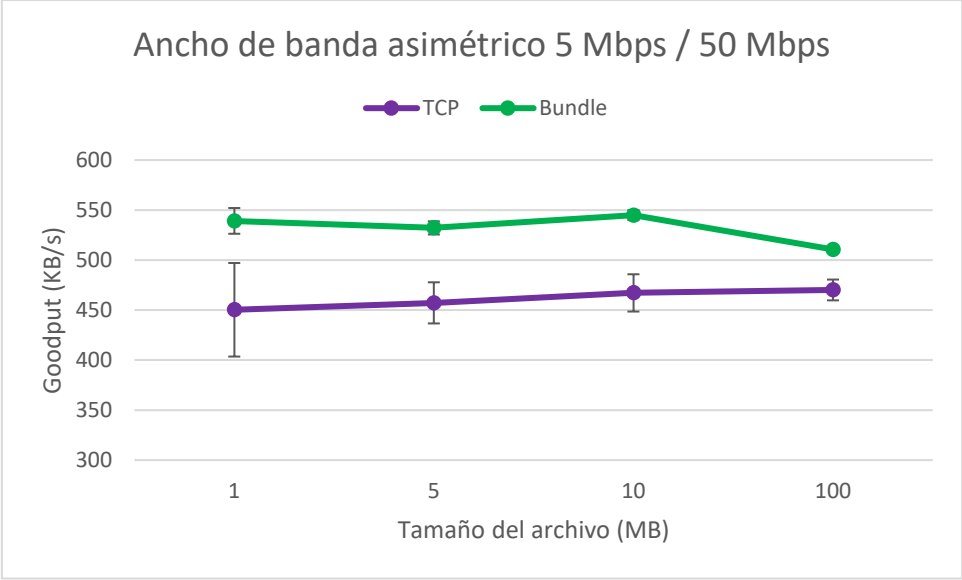

Figura 32. Valores de goodput con ancho de banda 5 Mbps / 50 Mbps

	Goodput (kB/s)		Desviación típica (σ)		Intervalo de confianza (90%)	
	Bundle	**TCP**	**Bundle**	**TCP**	**Bundle**	**TCP**
1 MB	539,143	450,274848	24,7557578	89,9091267	12,8766675	46,7660873
5 MB	532,1401	457,18084	12,6861213	39,5307936	6,59866556	20,5618786
10 MB	544,8690	467,1578	9,56200734	35,8256419	4,9736627	18,6346499
100 MB	510,5897	470,107672	5,82658902	19,9881775	3,03069088	10,3968167

Tabla 11. Valores del escenario con ancho de banda 5 Mbps / 50 Mbps

El escenario con ancho de banda asimétrico es muy común en el ámbito espacial ya que se tienen anchos de banda para enviar los paquetes y para recibir los ACKs. En este escenario, se puede comprobar como el protocolo Bundle tiene unos valores de goodput mayores y por lo tanto resulta más adecuado para este tipo de situaciones.

Para este escenario, se han tomado medidas de la tasa de transferencia de un archivo de 10 MB entre los dos routers igual que en el escenario anterior. Estas medidas se han realizado en el escenario con ancho de banda asimétrico de 5 Mbps / 50 Mbps y se pueden observar en las figuras 33 y 34.

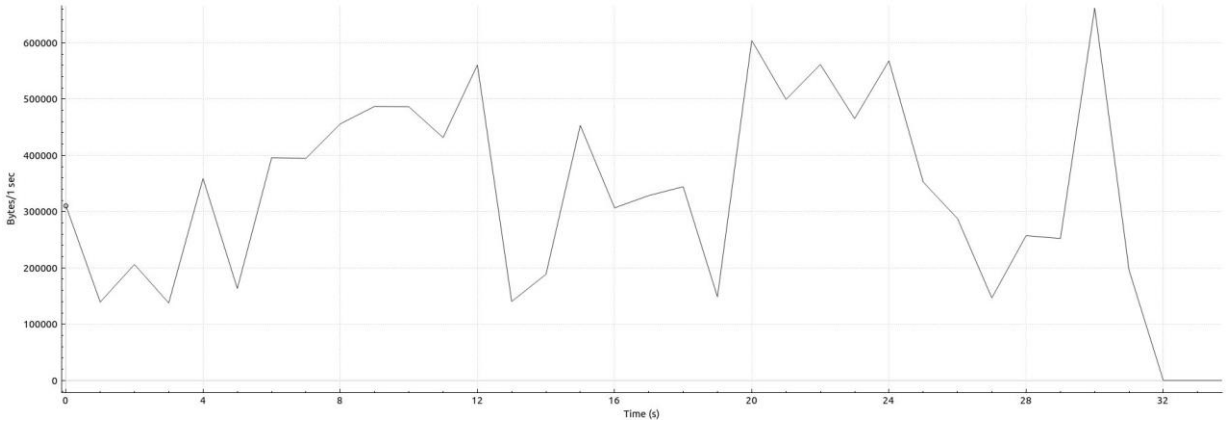

Figura 33. Gráfica de throughput con TCP en escenario de ancho de banda asimétrico

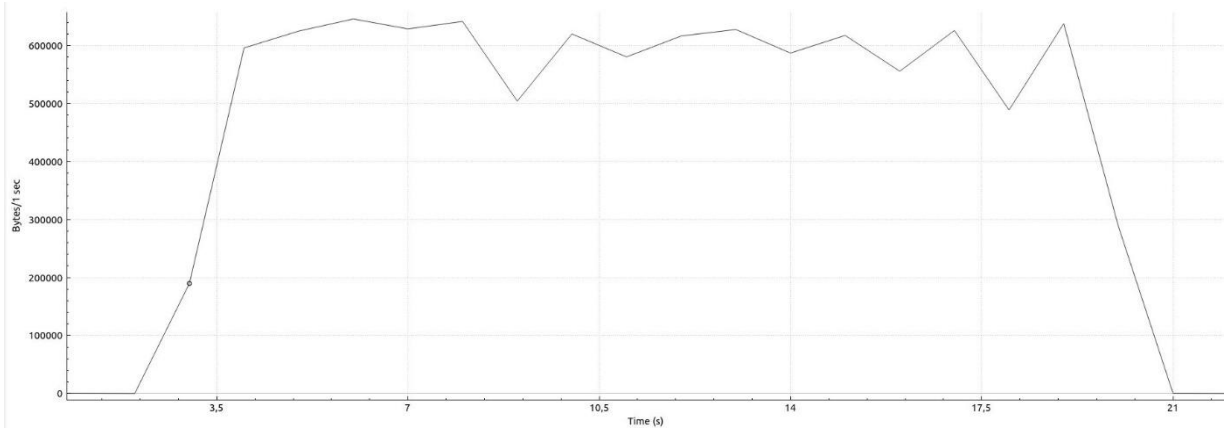

Figura 34. Gráfica de throughput con Bundle en escenario de ancho de banda asimétrico

En la figura 33 referida al comportamiento del protocolo TCP en este escenario, se puede observar cómo tiene una tasa de transferencia muy irregular con grandes caídas en comparación con la figura 34 donde la tasa de transferencia del protocolo Bundle es muy estable a lo largo del tiempo.

El distinto funcionamiento de ambos protocolos en este escenario se debe a que en TCP tenemos congestión, y el lado emisor está siempre adaptándose a intentar encontrar el ancho de banda disponible, por eso el throughput sube y baja con las pérdidas que se producen por la congestión en el segundo enlace. En bundle, la conexión TCP es en cada enlace, por lo tanto el primer router acumula los datos y los envía por la conexión TCP al segundo, no hay congestión porque el emisor ve directamente la capacidad del enlace (es el SO el que no deja al nivel bundle enviar más hasta que no ha conseguido enviar lo anterior), por eso es posible ajustarse al ancho de banda disponible de manera estable.

3.4.5 Escenario con ruptura de enlace

En este caso, se modificó la estructura de la red como se observa en la figura 27. Para este escenario se utilizaron dos routers unidos de forma inalámbrica con ancho de banda 100 Mbps y 20 ms de retardo, y dos PCs conectados a cada router a través de un switch. Para la realización de estas pruebas, primero se tomaron medidas del rendimiento de ambos protocolos en el escenario sin caídas de enlace y posteriormente, se rompió el enlace en un caso durante 20 segundos una vez establecida la comunicación y en otro caso durante 15 minutos también después de haber comenzado la comunicación.

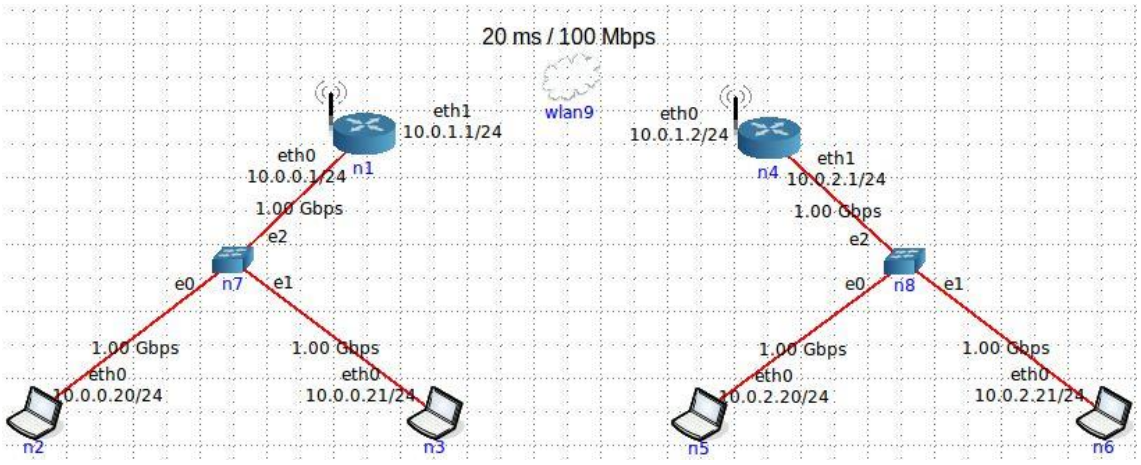

Figura 35. Escenario con ruptura de enlace

- Sin ruptura de enlace

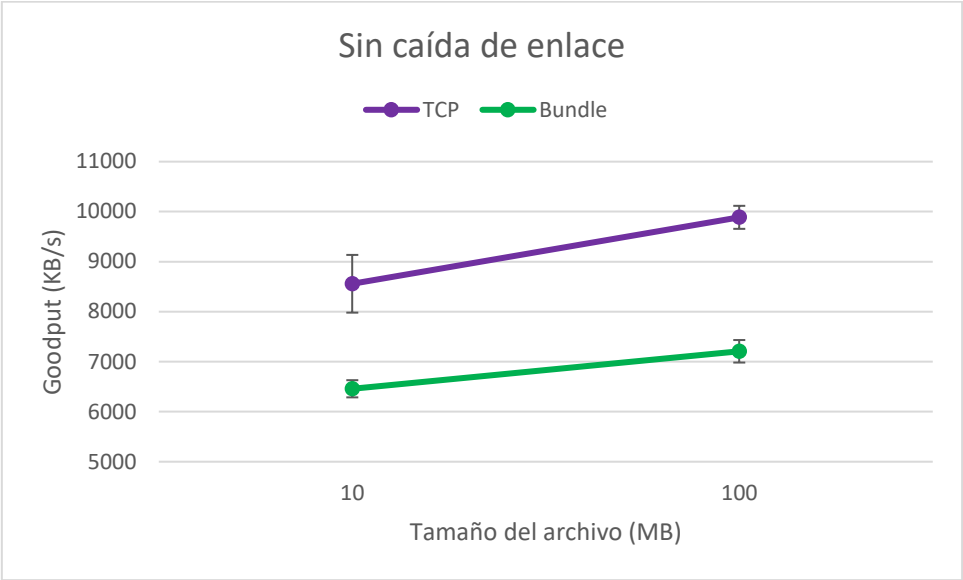

Figura 36. Valores de goodput sin ruptura de enlace

	Goodput (kB/s)		Desviación típica (σ)		Intervalo de confianza (90%)	
	Bundle	**TCP**	**Bundle**	**TCP**	**Bundle**	**TCP**
10 MB	6458,30531	8557,94595	330,440078	1109,61935	171,877874	577,166717
100 MB	7207,76016	9886,28813	433,21562	442,365369	225,336406	230,095633

Tabla 12. Valores del escenario sin ruptura de enlace

- Ruptura de enlace durante 20 segundos

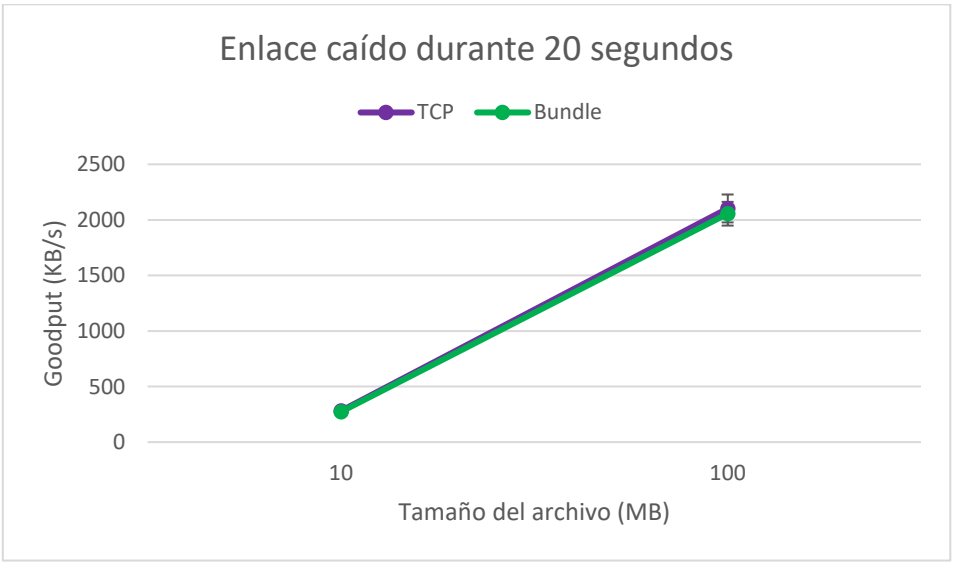

Figura 37. Valores de goodput con ruptura de enlace 20 segundos

	Goodput (kB/s)		Desviación típica (σ)		Intervalo de confianza (90%)	
	Bundle	**TCP**	**Bundle**	**TCP**	**Bundle**	**TCP**
10 MB	272,815674	280,795353	35,8324903	3,06561515	18,6382121	1,59457478
100 MB	2055,30431	2102,88853	215,027333	216,329318	111,846121	112,523346

Tabla 13. Valores del escenario con ruptura de enlace 20 segundos

- Ruptura de enlace durante 15 minutos

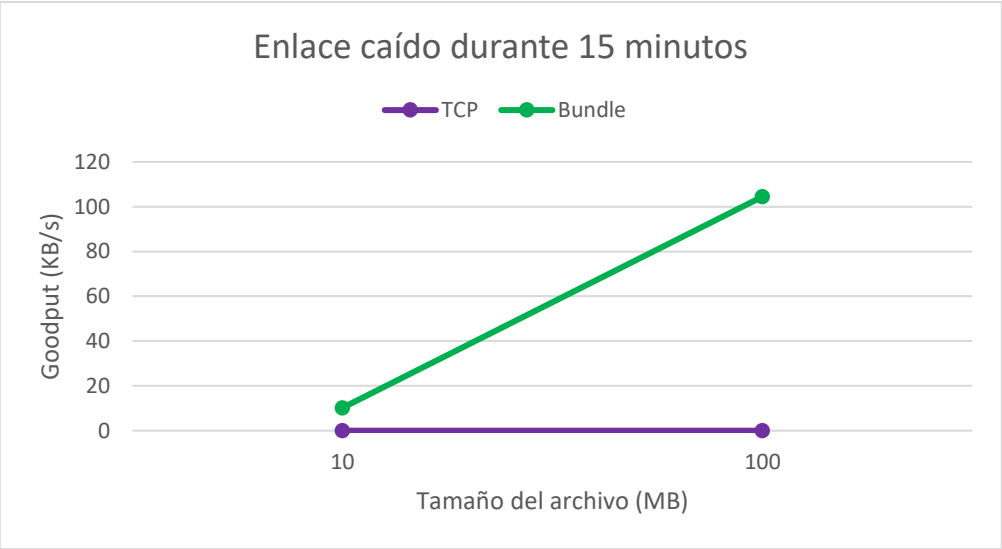

Figura 38. Valores de goodput con ruptura de enlace 15 minutos

	Goodput (kB/s)		Desviación típica (σ)		Intervalo de confianza (90%)	
	Bundle	**TCP**	**Bundle**	**TCP**	**Bundle**	**TCP**
10 MB	10,196541	---------	0,0255086	---------	0,01326826	---------
100 MB	104,496335	---------	1,51085751	---------	0,7858701	---------

Tabla 14. Valores del escenario con ruptura de enlace 15 minutos

En los escenarios de ruptura de enlace, se puede observar cómo TCP tiene un mejor funcionamiento cuando la red tiene un comportamiento normal sin rupturas de enlace. Sin embargo, cuando durante la transmisión se rompe el enlace durante 20 segundos, los valores de goodput en ambos casos, son prácticamente iguales. Este hecho se produce, ya que el periodo de tiempo con el enlace caído no es demasiado grande y TCP aún puede soportarlo de igual forma que Bundle. No obstante, justo cuando el periodo con el enlace caído alcanza los 15 minutos, la comunicación con el protocolo TCP es imposible ya que se supera el número máximo de retransmisiones y se asimila que el enlace esta caído y por lo tanto se finaliza la conexión. En el caso estudiado, debido a que el protocolo Bundle funciona sobre la capa de convergencia TCPCL, tiene el mismo problema pero sin embargo es capaz de abrir la conexión de nuevo y consigue realizar sin problema la transmisión una vez solucionada la caída del enlace ya que se encuentra

preparado para funcionar a pesar de caídas de enlace durante espacios temporales muy grandes. Para realizar un mejor estudio de este escenario, aunque no se han recogido los valores de goodput, también se realizaron pruebas con el enlace caído durante un periodo de 30 minutos y se pudo comprobar que el protocolo Bundle conseguía reactivar la conexión y realizar el envío del archivo con éxito. En este tipo de escenarios, es importante remarcar la existencia del almacenamiento persistente del que hace uso el protocolo Bundle, ya que permite suspender la transmisión y almacenar los paquetes durante la interrupción de la conexión para posteriormente reanudar la transmisión desde el mismo punto en el que se había dejado sin perder ningún paquete.

3.5 RESUMEN DEL CAPÍTULO

En este capítulo, inicialmente se ha explicado cual es el entorno que se ha utilizado para realizar las pruebas describiendo las características del emulador de redes CORE y de la implementación del protocolo Bundle ejecutada en cada nodo del emulador. Posteriormente, se han realizado y evaluado las pruebas en distintos escenarios y tras observar el comportamiento de ambos protocolos se puede determinar que el protocolo TCP tiene un rendimiento muy superior en un escenario en el que las condiciones sean las de una red tradicional con escasos retardos, porcentajes de pérdidas muy bajos y rupturas de enlaces inexistentes o durante cortos periodos tiempo. Sin embargo, el protocolo Bundle tiene mejor rendimiento cuando se dan situaciones similares a las de una DTN con retardos muy grandes, altos porcentajes de pérdidas y rupturas de enlaces durante largos periodos de tiempo.

En el siguiente capítulo, se realizará un análisis de la planificación y el presupuesto requerido para la realización de las pruebas y del trabajo en general.

CAPÍTULO 4.

GESTIÓN Y DESARROLLO DEL PROYECTO

En este capítulo se realiza un análisis de cuál ha sido la planificación temporal que ha seguido el proyecto y cuál ha sido el presupuesto necesario para su consecución.

4.1 PLANIFICACIÓN DEL PROYECTO

Según la "Guía de los fundamentos para la dirección de proyectos" [38], se define un proyecto como "un esfuerzo que se lleva a cabo para crear un producto, servicio o resultado único, y tiene la característica de ser naturalmente temporal, es decir, que tiene un inicio y un final establecidos". En este caso, el proyecto comienza con la búsqueda del tutor y finaliza con la sesión de defensa. Durante el transcurso del proyecto se han identificado las siguientes tareas:

A. Planificación y organización: desde el día 17 de octubre al 31 de octubre se realizó un análisis de las necesidades del proyecto y cuál sería su distribución a lo largo del tiempo. Se dedicaron 15 horas.
B. Documentación y aprendizaje teórico: se trata de una de las labores más importantes del proyecto ya que gran parte del trabajo se basa en esta tarea. Se prolongó desde el 1 de noviembre hasta el 31 de enero de 2019. Se dedicaron 60 horas.
C. Aprendizaje e instalación de CORE: se realizó desde el día 15 de enero al día 28 de febrero. Consistió en la lectura de documentación sobre el emulador CORE, que permitió la instalación y el conocimiento de todas las funciones que posee el programa. Se dedicaron 35 horas.
D. Realización de las pruebas en CORE: desde el día 1 de marzo al día 30 de abril se realizaron las distintas pruebas previstas en el emulador CORE. Se dedicaron 60 horas.
E. Obtención de resultados y conclusiones: tuvo lugar desde el día 1 de abril hasta el 15 de mayo. Durante la realización de las pruebas se fueron generando resultados que finalmente han sido expuestos y evaluados en esta memoria. Se dedicaron 40 horas.

F. Redacción de la memoria: comenzó el día 1 de diciembre y finalizó el día 31 de mayo en el que se dio por terminada la escritura de la memoria. Se dedicaron 110 horas.

En la figura 31 se puede observar un diagrama de Gantt para obtener una visión grafica de cual ha sido la distribución de las tareas en el tiempo.

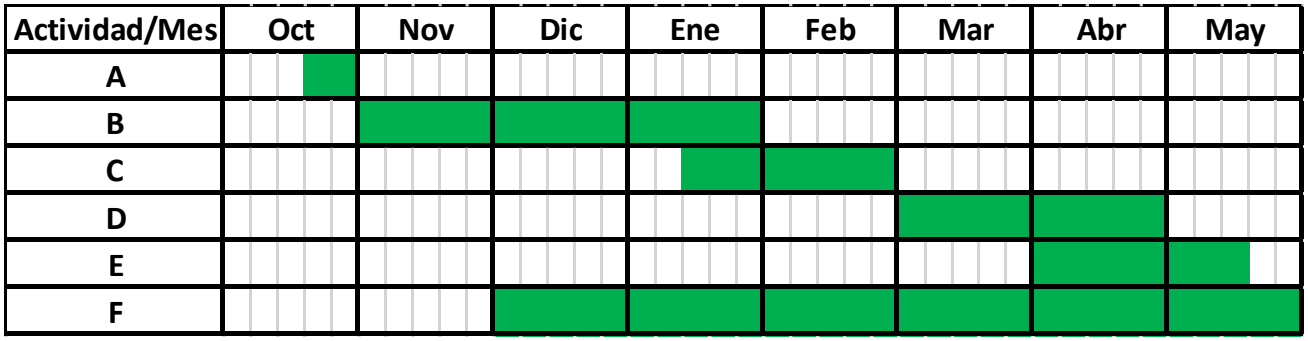

Figura 39. Diagrama de Gantt

4.2 PRESUPUESTO

El apartado económico es uno de los más importantes a la hora de realizar un proyecto. En este caso, se ha realizado un análisis de cuál ha sido el presupuesto necesario para realizar este proyecto y para ello se ha separado en dos partes:

- Costes directos: como se observa en las tablas 15 y 16, comprenden los recursos tanto de personal como de material empleados en el proyecto.

Puesto	Tiempo trabajado	Coste por hora	Coste total
Ingeniero Senior	30 horas	40 €	1.200 €
Ingeniero Junior	320 horas	25 €	8.000 €
		TOTAL	9.200 €

Tabla 15. Costes del personal

Concepto	Coste	Dedicación	Periodo depreciación	Coste imputable
Ordenador portátil HP	1200 €	8 meses	60 meses	160 €
Licencia Microsoft Office	70 €	8 meses	60 meses	9,33 €
			TOTAL	169,33 €

Tabla 16. Costes del material

- Costes indirectos: comprenden los gastos necesarios para la realización del proyecto pero que no se pueden asignar concretamente a ningún producto. Este tipo de costes incluye entre otros conceptos, la electricidad empleada, la conexión a internet o la gasolina consumida en los trayectos entre la universidad y mi casa. Para realizar una estimación de estos costes, se ha calculado en un 15% de los costes directos, siendo por lo tanto de 1.405,4 €.

Finalmente teniendo en cuenta los dos tipos de costes se puede comprobar en la tabla 17 cual ha sido el coste total del proyecto.

Concepto	Coste
Coste de personal	9.200 €
Coste de material	169,33 €
Costes indirectos	1.405,4 €
TOTAL	10.774,73 €

Tabla 17. Costes totales

A continuación, en la tabla 18 se adjunta un presupuesto formalizado recogiendo la información presentada anteriormente.

UNIVERSIDAD CARLOS III DE MADRID
Escuela Politécnica Superior

PRESUPUESTO DE PROYECTO

1.- Autor: Carlos Díez Rodríguez

2.- Departamento: Ingeniería Telemática

3.- Descripción del Proyecto:

- **Titulo**: Evaluación del protocolo Bundle mediante un emulador de redes
- **Duración (meses):** 10 - Tasa de costes Indirectos: **15%**

4.- Presupuesto total del Proyecto (valores en Euros): 10.774,73 €

5.- Desglose presupuestario (costes directos)

COSTE PERSONAL

Apellidos y nombre	Categoría	Dedicación (horas)	Coste persona mes	Coste (Euro)
Soto Campos, Ignacio	Ingeniero Senior	30	40,00	1.200,00
Díez Rodríguez, Carlos	Ingeniero	320	25,00	8.000,00
			Total	9.200,00

COSTE MATERIAL

Descripción	Coste (Euro)	% Uso dedicado proyecto	Dedicación (meses)	Periodo de depreciación	Coste imputable
Ordenador portátil HP	1.200,00	100	8	60	160,00
Licencia Microsoft Office	70,00	100	8	60	9,33
				Total	169,33

6.- Resumen de costes

Presupuesto Costes Totales	**Presupuesto Costes Totales**
Personal	7.600
Amortización	169,33
Costes Indirectos	1.405,4
Total	10.774,73

Tabla 18. Presupuesto

CAPÍTULO 5.

CONCLUSIONES Y LÍNEAS FUTURAS

Tras la realización de este trabajo, se ha podido comprobar la importancia que tiene y tendrá en un futuro, el protocolo Bundle ya que posee un gran rango de aplicaciones en el entorno de las redes tolerantes al retardo y a las interrupciones. Durante el capítulo 2, se han mostrado las características tanto del protocolo Bundle como del tipo de redes en el que opera. Además, gracias a la utilización del emulador de redes CORE, se ha podido verificar experimentalmente en un escenario real, cómo en la actualidad, a pesar de ser un protocolo relativamente reciente, tiene un rendimiento superior al del protocolo TCP en determinados escenarios. En concreto, en escenarios con largos retardos, escenarios con altos porcentajes de pérdida de paquetes, escenarios con enlaces con ancho de banda asimétrico y escenarios con ruptura de enlaces durante largos periodos de tiempo. Por otro lado, también se ha validado el entorno basado en CORE como un mecanismo flexible y apropiado para hacer experimentación sobre el protocolo Bundle.

Al ser un protocolo que se encuentra aún en fase experimental, posee la capacidad de variar algunas de sus características para poder adaptarse a las necesidades que requiera cada entorno. Entre los principales puntos sobre los que se podría trabajar para mejorar su rendimiento están:

- Hacer un uso más eficiente de las baterías de los nodos, gestionando de mejor forma la energía ya que es un elemento imprescindible para que se pueda establecer la comunicación. En una DTN, debido a los entornos extremos en los que se suelen situar los nodos es muy complicado realizar una sustitución o recarga de la batería y por ello se deben desarrollar métodos de ahorro de energía. Sobre este punto, se pueden desarrollar modelos de predicción sobre la movilidad de los nodos de manera que se pueda conocer con anterioridad cuando se van a producir los contactos y de esta manera, no activar el nodo hasta que no sea necesario.
- Realizar un mejor uso del espacio de búfer en los nodos. Debido a que algunos tipos de enrutamiento como el epidémico, realizan una cantidad ilimitada de

réplicas que se almacenan en todos los nodos, en muchas ocasiones se llenan completamente de forma innecesaria.

Por otro lado, debido a que el protocolo tiene diversas implementaciones como DTN2 e ION-DTN y diversas capas de convergencia como LTP y Saratoga, se podría realizar en un proyecto futuro una comparativa entre todos ellos para comprobar cuál sería la configuración más adecuada en cada tipo de escenario.

Finalmente, aunque en este trabajo se han validado experimentalmente las propiedades del protocolo Bundle en redes DTN, el comportamiento en algunos escenarios es complejo de analizar y requiere mayor estudio en el futuro.

ANEXO A.

SUMMARY

1. INTRODUCTION

Nowadays, communication networks are a basic element in the life of people. Since the beginning of time means of communication have evolved to the present day in which we can communicate with any person in the world, even from outer space, in a very small length of time. A large part of the communications that are carried out for the transfer of data between computers is done thanks to the family of TCP / IP protocols that form the basis of what is known as the Internet. However, at present, the situations in which it is desired to establish communication with another device can be very varied and in many extreme cases, it represents a challenge for the network to make communication possible. In these situations, the need arises to create networks that can cope with the challenges that the traditional network could not support and that is when Delay and Disruption Tolerant Networks (DTN) appear. This type of networks is capable of communicating with a greater degree of reliability thanks to the Bundle protocol studied in this project.

2. STATE OF ART

2.1 Delay and Disruption Tolerant Networks

The Delay and Disruption Tolerant Network is a network architecture consisting of several nodes that send and receive packets, called bundles. This type of network uses the Bundle protocol as a rule to establish communication between the nodes. DTN arise from the need to overcome the challenges of establishing and maintaining communication in an extreme situation such as communications with outer space, communications in rural areas or communications in an aquatic environment. All its properties are included in RFC 4838 and this kind of network is characterized by:

- Support intermittent connections since in a DTN, the nodes that form it have limited energy and to save energy they close their links periodically so the topology changes. In addition, the density of the nodes is usually smaller and more distant than in a traditional network.

- Ability to assimilate high delays and low efficiencies. The end to end delay is the sum of the total delay of each hop on the route. The delay in each hop is constituted by the waiting time, the waiting time in queue and the transmission time. The delay in each hop can be very high due to the distance to the destination and the possibility that the connection is intermittent and there are changes in the topology. On the other hand, due to the frequent fragmentations that are made in a DTN, the waiting time in queue can also be increased.
- Persistent storage capacity due to intermittent connections. It is necessary that the data can be stored indefinitely in a node until the next hop is available. In addition, in case of communication errors it is also necessary that the data is stored in the origin node since it would have to be forwarded. In this type of network, if the storage space was like in a traditional network, the packet loss rate would be very high.
- Very weak security due to the lack of a specialized service for this and the limited experimentation possible until now in real scenarios.

The architecture of a DTN, according to RFC 4838, is formed by regions that are each of the networks that have homogeneous communication characteristics and that form the DTN. Each region is formed by nodes. Depending on the use the nodes make of the Bundle protocol to send or receive (or both) bundles, they will be called in a different way. A host is a node which can only send or receive bundles but without forwarding capacity. A router allows the forwarding of packets within the region and can sometimes perform host functions. The third kind of node is a gateway which act as an intermediary between two DTN regions and can sometimes perform host functions.

In a DTN, since the nodes are not always active or within the transmission range of the sending node, the connections are intermittent. The moment in which communication takes place between two nodes is called contact and according to RFC 4838 there are five types, opportunistic, scheduled, predicted, persistent and on demand.

Once the contact between the two nodes has taken place, an exchange of the necessary information about the surroundings is carried out and the bundles are sent. Depending on how the communication is made, in a DTN there are several types of routing. Among them we can mention: Direct Delivery, First Contact, Epidemic, Spray and Wait, Fuzzy

Spray, Probabilistic Routing Protocol using History of Encounters and Transitivity (PRoPHET), Sensor Context Aware Routing (SCAR), FAD, MaxProp, Resource Allocation Protocol for Intentional DTN (RAPID), Cluster-Based Routing, NECTAR, Opportunistic DTN Routing with Window-Aware Adaptive Replication (ORWAR), History Based Routing Protocol for Opportunistic Networks (HiBOp), BubbleRap and SimBet.

Over the years, thanks to the research that has been conducted to learn more about DTNs, numerous situations have been discovered in which their use is more effective than traditional networks. Therefore, this type of network architecture has increased its appearance in two environments mainly:

- Interplanetary networks: it was the environment where the first networks of this type were developed. Due to their special characteristics, the protocols of the traditional network were not effective. In this environment, the National Aeronautics and Space Administration (NASA) was the first to incorporate the DTN network architecture to increase reliability in communications and currently has three communication networks that are the Deep Space Network (DSN), Near Earth Network (NEN) and the Space Network (SN) used by NASA and other space agencies. Some of the most important projects in which NASA has introduced the DTN architecture and the Bundle protocol have been Deep Impact Networking (DINET) experiment, the Earth Observing Mission 1 (EO-1) and the Lunar Laser Communication Demonstration (LLCD).
- Terrestrial networks: in this environment, the importance of DTNs has been growing very fast due to the useful applications that can have in different areas such as:
    - Military networks: in this environment, due to its characteristics (high mobility of the nodes, interference by orography, limited availability of spectrum or the introduction of interference by the enemy) it is very necessary to apply the architecture of a DTN. Therefore, the United States Marine Corps Army is developing the CONDOR project with the objective of uniting tactical data networks that until now were difficult to unite. However, in order to be truly effective, a DTN in this environment must improve the aspect of security since the data they handle are very

sensitive and therefore they are working together with the DTN Research Group (DTNRG) to develop more reliable communications.

- Networks for the control of nature: in this aspect, the DTN architecture has been used in projects dedicated for obtaining a faster and more effective response to natural disasters thanks to seismic monitoring and fire control. On the other hand, due to the current situation, numerous projects are also being developed in defense of the environment. Some of these projects are, ZebraNet which consists of monitoring the habits and behaviour of zebras by installing a global positioning collar (GPS), SWIM which is a project similar to ZebraNet but in this case with the activity of the whales and Environmental Monitoring in Metropolitan Areas (EMMA) which is a project created to control pollution levels in metropolitan areas.
- Networks in rural areas, underdeveloped or in remote environments: in these environments due to the type of terrain, the distances to large cities or the economic situation of the country, it is very complicated to get the connection to the Internet. Therefore, to improve the situation in this type of environment, the DTN architecture has been integrated in some projects like DakNet which is a project developed to give connection to rural villages of India and Cambodia; Wizzy Digital Courier, which is similar project developed in some schools in South Africa; and N4C, a project that aims to track reindeer and obtain meteorological and environmental data in remote areas of Lapland and in the mountainous region of Kočevje in Slovenia for being used by hikers.
- Vehicle networks: they aim to increase safety and reduce the number of accidents. Its structure is based in a DTN with two types of nodes. Stationary relay stations (SRS) act as fixed nodes located along roads and highways and usually without an Internet connection and without the possibility of communicating with each other. On the other hand, the vehicles enabled with this functionality act as mobile nodes connecting and exchanging information with the SRS. Thanks to this type of networks, information about traffic congestion and parking availability

can be disseminated, and they can also provide basic connectivity for passengers of public transportation vehicles.

2.2 Bundle Protocol

According to the Internet Research Task Force (IRTF) the nodes that make up a DTN must use the Bundle protocol. The packets that are sent through this protocol are called bundles. This protocol, whose characteristics and properties are included in RFC 5050, has been chosen as the main one in a DTN because it has a storage and forwarding system and a custody transfer mechanism that make possible a more reliable delivery of the information in extreme environments. Due to the nature of this type of networks that have an intermittent connection, the storage and forwarding system of the Bundle protocol sends the information from the storage unit of a node, to the storage unit of the destination node. Since both have a persistent storage unit, the information can be stored for long periods of time until the sending node has the opportunity to establish contact again with another node and forward the information. On the other hand, the custody transfer mechanism allows the sending node, once it has made the shipment, to delegate the responsibility of the shipment to the destination node transferring custody and recovering the resources that were being consumed. This mechanism may not be implemented in all the nodes since it must meet a series of requirements:

- Availability for storage over a long period of time
- Forwarding capacity with the objective of delivering the bundle to the final destination
- Enough energy to stay active for a long period of time
- Ability to work cooperatively with the rest of the nodes taking advantage of all the opportunities to send the bundle.

One of the weaknesses of this mechanism is that it does not have any system to detect errors or reject damaged packages.

According to RFC 5050, the bundles generated by the Bundle protocol must be the union of at least two block structures. The bundle must have a primary header block, an optional payload block and a set of optional extension blocks that will follow the primary block, such as the payload security block (PSB). To determine the length of

the bundle, the last block of the sequence must have the indicator bit of the last block to 1. In addition, some fields of the primary block implement the encoding scheme Self-Delimiting Numeric Values (SDNV) in order to make a minimum consumption of the bandwidth and it allows the protocol to have flexibility to face new requirements in the future.

In many cases the packets must cross very heterogeneous networks with different limitations in the capacity of the links, so the Bundle protocol has support for fragmentation of the bundles. In this way, the communication efficiency between the DTN nodes is improved, guaranteeing the full use of the link when the contact occurs and avoiding wasting resources in sending packets that have been partially sent. There are two types of fragmentation according to the moment in which it is carried out, proactive fragmentation and reactive fragmentation.

The Bundle protocol requires the protocols of the underlying layers to carry out the transport of the bundles successfully. Therefore, it maintains communication with these layers through the convergence layers. In this way, when using the convergence layer adapter service, the Bundle protocol can perform communications using the most appropriate packet transport protocol for each type of network, adapting to its characteristics. Some of the most used convergence layers are:

- Licklider Transmission Protocol (LTP): it was designed for use in the field of outer space joining long distances or point to point links characterized by a very large transmission delay and with the possibility of interruptions in the connection. Its characteristics are included in RFC 5326 where it is specified that it has a mechanism for dividing the data into two parts according to the degree of reliability in the delivery. For scenarios with several jumps in which more complex reliability systems are required, it has served as the basis of the Licklider Transmission Protocol Transport (LTP-T).
- Saratoga Protocol: it is a light protocol based on the User Datagram Protocol (UDP). It is widely used in environments with private connection links that have an asymmetric transmission rate or a unidirectional communication channel since this protocol makes a very efficient use of contact time by completely filling the link.

- UDPCL: it is a little recommended convergence layer, because very restrictive requirements must be met to make the Bundle protocol work on the UDP protocol
- TCPCL: it is characterized by using the Transmission Control Protocol (TCP) as a transport layer. Its characteristics are included in RFC 7242.

Regarding the security, it has been considered during the development of the protocol and therefore the Bundle Security Protocol has been created. Thanks to this protocol, the use of the network is denied to unauthorized nodes avoiding the consumption of resources and guaranteeing the integrity of the data that is sent and received, ensuring confidentiality within the network. To do this, you can add three types of blocks in each package depending on the policies and requirements of the network. These blocks are the Bundle Authentication Block (BAB), the Payload Security Block (PSB) and the Payload Confidentiality Block (PCB).

3. BUNDLE PROTOCOL IN A NETWORK EMULATOR

3.1 CORE emulator

The CORE network emulator is an open source program initially developed by Boeing and currently supported by the United States Naval Research Laboratory. CORE provides a graphical user interface (GUI) allowing to create fixed or mobile networks and generate traffic between them in real time. It also offers some useful functions for the user allowing to know the traffic that runs through the links or modify what are the conditions of the links establishing different values of delay, bandwidth ...

There are several network emulators today, but CORE has been chosen for the implementing the project due to the previous knowledge that the Telematic Engineering Department of the Carlos III University of Madrid have and the ease of use of CORE in order to reuse the scenarios and results obtained in later studies. In addition, it allows the implementation of different protocols in each scenario, which in this case is very useful to establish comparisons between the performance of the Bundle and TCP protocol.

3.2 Testing and results

Install one of the implementations of Bundle protocol is necessary to use it in CORE emulator Some of the most important implementations are DTN2, ION-DTN and IBR-DTN, which has been chosen for the tests since it is a portable and very light implementation. The tests have been carried out in the following environments:

- Scenario with long delays
- Scenarios with a high percentage of losses
- Scenario with asymmetric bandwidth
- Scenario with broken links

In all these scenarios, it has been possible to verify that due to its characteristics, the Bundle protocol has a better performance offering higher goodput values than the TCP protocol.

4. CONCLUSIONS AND FUTURE DEVELOPMENT

After carrying out this work, it has been possible to verify the importance that the Bundle protocol has and will have in the future as it has a wide range of applications in the environment of Delay and Disruption Tolerant Networks. During chapter 2, the characteristics of the Bundle protocol and the type of networks in which it operates are shown. In addition, thanks to the use of the CORE network emulator, it has been possible to verify in a real scenario that it has a higher performance than the TCP protocol in certain environments. Specifically, in environments with long delays, environments with high percentages of packet loss, environments with links with asymmetric bandwidth and environments with broken links for long periods of time.

The objective of this work is to offer greater knowledge about the different possibilities offered by the Bundle protocol and thereby increase the interest to improve its characteristics. Bundle protocol is still in experimental phase so it has the ability to vary some of its characteristics to be able to adapt to the needs required by each environment. Among the main points on which efficiency could be improved are:

- Making a use more efficient of the node's batteries, due to the management of energy is an essential element for stablishing communication.

- Make better use of the buffer in the nodes because some types of routing, such as the epidemic, perform an unlimited number of replicas that are stored in all the nodes so in many cases they are filled completely unnecessarily.

ANEXO B.

CONFIGURACIÓN DE CORE Y IBR-DTN

Para la realización de las pruebas, se ha elegido el emulador de redes CORE [34] y para su instalación en el sistema operativo Linux Ubuntu se han seguido los siguientes pasos recogidos en su página web [34]:

- Para instalar CORE:

```
apt-get install core-network
```

- Para la instalación de diversos complementos para el emulador:

```
sudo apt-get update sudo apt-get dist-upgrade sudo apt-get install bash bridge-utils
ebtables iproute libev-dev python tcl8.5 tk8.5 libtk-img
```

Por otro lado, IBR-DTN es la implementación del protocolo Bundle utilizada para la realización de las pruebas y cuyo rendimiento se ha podido comprobar en comparación con el de TCP.

Para poder usar esta implementación dentro del emulador CORE, ha sido necesaria su instalación en Linux siguiendo las instrucciones indicadas en la página de IBR-DTN [39]. Debido a que el sistema operativo de la máquina virtual utilizado ha sido Linux Ubuntu, los comandos a ejecutar para instalar la implementación han sido:

- Para añadir el repositorio:

```
wget -O - http://download.opensuse.org/repositories/home:/j_morgenroth/
xUbuntu_14.04/Release.key | \ sudo apt-key add -

deb http://download.opensuse.org/repositories/home:/j_morgenroth/xUbuntu_14.04 ./
```

- Para instalar IBR-DTN:

```
sudo apt-get update

sudo apt-get install ibrdtnd ibrdtn-tools
```

Una vez realizadas ambas instalaciones, para lanzar el emulador, se deben ejecutar los siguientes comandos:

```
sudo /etc/init.d/core-daemon start

core-gui
```

Una vez instalado, se podrán añadir diversos elementos como routers, PCs, switches utilizando la interfaz gráfica del emulador como se observa en la figura 32.

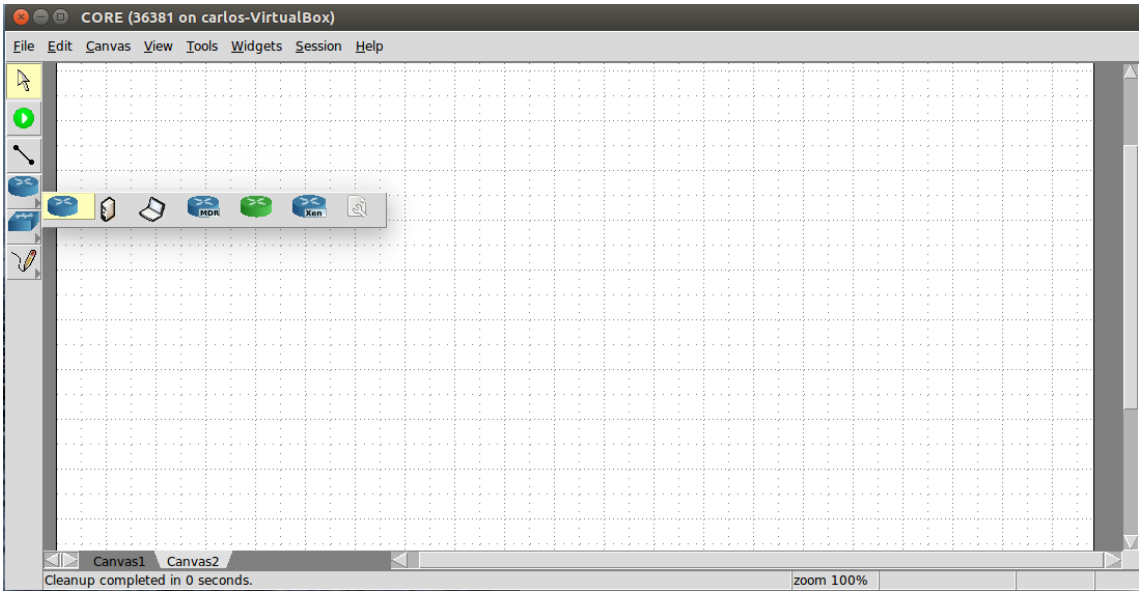

Figura 40. Interfaz gráfica de CORE

Posteriormente, tras implementar la estructura de red deseada, se pulsará el botón verde del panel izquierdo para comenzar la sesión. Durante la sesión, se pueden modificar distintos valores (ancho de banda, porcentaje de pérdidas, retardo…) en los enlaces haciendo doble click sobre ellos y también haciendo doble click sobre los distintos elementos de la red aparece una ventana que permite la configuración de distintos valores como direcciones IP, MAC o el rango de cobertura si creamos una red inalámbrica. Para ejecutar IBR-DTN se debe modificar el archivo de configuración creado por la implementación y se debe introducir en el directorio temporal que se crear al iniciar la sesión para cada nodo. Dicho archivo de configuración permite modificar distintos valores del protocolo Bundle. A continuación, se muestra el archivo de configuración con los principales parámetros que permite configurar y los valores que se han usado en el trabajo:

```
####################################
# IBR-DTN daemon                   #
####################################
#
#Permite modificar el EID del nodo, en este caso, no se ha modificado
#y se ha dejado por defecto
#
#local_uri = dtn://node.dtn

#
```

```
# Permite elegir el directorio donde se encuentra el log
#
logfile = /var/log/ibrdtn/ibrdtn.log

#
# Permite modificar el valor de los bundles
#
# The value accepts different multipliers.
# G = 1,000,000,000 bytes
# M = 1,000,000 bytes
# K = 1,000 bytes
#
#limit_blocksize = 1.3G


#
# Permite limitar el desplazamiento del timestamp, rechazando los
#valores que se encuentren por encima
#
#limit_predated_timestamp = 604800

#
# Limita el tiempo de vida máximo del bundle de forma que los bundles
# con un valor superior son rechazados
#
#limit_lifetime = 604800

#
# Limita el número de bundles que se encuentran en tránsito, por
#defecto es 5
#
#limit_bundles_in_transit = 5

#
# Permite habilitar soporte para fragmentación. Está habilitado por
#defecto.
#
#fragmentation = no

#
# Si la fragmentación está habilitada, permite limitar el tamaño de
#los fragmentos
#
# limit_payload = 500K

####################################
# storage configuration            #
####################################

#
# Permite definir una carpeta de almacenamiento temporal.
#
#blob_path = /tmp

#
# Permite definir una carpeta de almacenamiento persistente. En
#nuestro caso, este parámetro ha sido modificado creándose una carpeta
#en el directorio de cada nodo llamada stX, siendo X el número de nodo
#
storage_path = stX
```

```
#
#
# Permite limitar el tamaño máximo de almacenamiento. En nuestro caso,
se ha modificado este valor a 320 MB.
#
# G = 1,000,000,000 bytes
# M = 1,000,000 bytes
# K = 1,000 bytes
#
limit_storage = 320M


####################################
# convergence layer configuration   #
####################################

#
# discovery over UDP/IP

# Permite cambiar el método de descubrimiento siendo IPND versión 1 el
#establecido por defecto
# 0 = DTN2 compatible discovery
# 1 = IPND version 0
# 2 = IPND version 1 (default)
#discovery_version = 2

# Permite deshabilitar el anuncio de descubrimientos estableciendo
#este parámetro a 0. Está establecido en 1 por defecto.
#
#discovery_announce = 0

#
# Permite enumerar la lista de interfaces creadas
#
#net_interfaces = lan0

#
# Permite cambiar la configuración de la capa de convergencia elegida
para cada interfaz de cada nodo, en el ejemplo se observan TCP y UDP
#
#net_lan0_type = tcp           # we want to use TCP as protocol
#net_lan0_interface = eth0     # listen on interface eth0
#net_lan0_port = 4556          # with port 4556 (default)
#
#net_lan1_type = udp           # we want to use UDP as protocol
#net_lan1_interface = eth0     # listen on interface eth0
#net_lan1_port = 4556          # with port 4556 (default)

#
# TCP tuning options
#
# Con TCP los bundles son divididos en chunks, y con este parámetro
#se puede modificar su tamaño máximo. Está establecido en 4096 por
#defecto.
#
#tcp_chunksize = 4096
#
# Permite establecer un valor máximo del temporizador de TCP en el que
#se realice la desconexión del nodo. Por defecto este valor es 0 por
#lo que está deshabilitado.
#tcp_idle_timeout = 0
```

```
#
# Permite modificar la frecuencia con la que se emiten los paquetes
3KEEPALIVE con TCPCL. Por defecto es 60.
#
#keepalive_timeout = 60


####################################
# routing configuration            #
####################################

#
# Permite modificar el modo de enrutamiento entre los siguientes
#valores: default | epidemic | flooding | prophet | none
# Con el valor por defecto, el protocolo solo envía bundles a los
#nodos vecinos y a las rutas estáticas configuradas. En nuestro caso,
#el tipo de enrutamiento elegido fue PRoPHET que realiza el envío
#basándose en la probabilidad de que la comunicación se realice con
#éxito.
#
routing = prophet

#
# Permite habilitar el reenvío. Habilitado por defecto.
#
#routing_forwarding = yes


#
# Permite definir rutas estáticas con el siguiente patrón:
# - <target-scheme> <routing-node>
# Por ejemplo:
#
# route1 = ^dtn://n5 dtn://n1 dtn://n4

#
# static connections
# Permite configurar las rutas estáticas modificando los siguientes
#parámetros

### node-five.dtn ###
#static1_address = 10.0.0.5    # the node has the address 10.0.0.5
#static1_port = 4556                   # accept bundles on port 4556
#static1_uri = dtn://node-five.dtn     # eid of the node is
"dtn://node-five.dtn"
#static1_proto = tcp                   # reachable over TCP
#static1_immediately = yes     # connect immediately to this node
#static1_global = yes          # this node is only reachable with
internet access

### node-ten.dtn ###
#static2_address = 192.168.0.10        # the node has the address
10.0.0.10
#static2_port = 4556                   # accept bundles on port 4556
#static2_uri = dtn://node-ten.dtn      # eid of the node is
"dtn://node-ten.dtn"
#static2_proto = udp                   # reachable over UDP
#static2_immediately = no              # connect on-demand to this node

### node-fifteen.dtn ###
```

```
#static3_email = fifteen@example.com    # the email address of the
node
#static3_uri = dtn://node-fifteen.dtn   # eid of the node is
"dtn://node-fifteen.dtn"
#static3_proto = email                 # reachable over MCL

### prophet configuration ###
# Permite modificar distintos valores de PRoPHET, que en nuestro caso
#se han mantenido los que estaban por defecto.
#
#prophet_p_encounter_max = 0.7
#affects how strong the predictability is increased on an encounter

#prophet_p_encounter_first = 0.5
#the predictability of a neighbor on the first encounter

#prophet_p_first_threshold = 0.1
#lowest predictability when neighbors predictabilities are forgotten

#prophet_beta = 0.9
#Weight of the transitive property

#prophet_gamma = 0.999
#Determines how quickly predictabilities age

#prophet_delta = 0.01
#(1-delta) is the maximum predictability

#prophet_time_unit = 1
#time unit in seconds

#prophet_i_typ = 300
#typical time interval between two node encounters

#prophet_next_exchange_timeout = 600
#timeout how often handshakes should be executed

#prophet_forwarding_strategy = GRTR
#The forwarding strategy used GRTR | GTMX

#prophet_gtmx_nf_max = 30
#Maximum times to forward in the GTMX strategy

#prophet_push_notification = no
#Push notifications to neighbours if new routes are found
```

También se pueden modificar más parámetros relacionados con el protocolo de seguridad de Bundle, sobre el servicio de nombre DHT y sobre la capa de convergencia email, pero no han sido relevantes para el desarrollo del trabajo.

Una vez modificado el archivo de configuración de forma específica para cada nodo, se debe copiar en el directorio de cada nodo y posteriormente ejecutarlo, lanzando un terminal haciendo doble click en cada nodo. Para ello, se debe ejecutar el siguiente comando:

```
user@n2:~$ dtnd -i eth0 -c i2.conf
```

En este caso, se estaría implementando el protocolo Bundle modificado en el archivo de configuración i2.conf dentro del nodo n2 en su interfaz eth0. Esta ejecución se deberá realizar para todos los nodos en los que se desee utilizar el protocolo Bundle, tanto para enviar como para recibir.

Finalmente, algunos de los comandos de IBR-DTN que se han utilizado son han sido:

- Dtnping: que nos permite realizar un ping a través del protocolo Bundle para comprobar la conexión

```
user@n2:~$ dtnping dtn://n1/echo
```

- Dtnsend: nos permite enviar un archivo hacia otro nodo.

```
user@n2:~$ dtnsend dtn://n5/st5 nombreArchivo1
```

BIBLIOGRAFÍA

[1] J. Postel, «RFC 1543 Instructions to RFC Authors,» October 1993. [En línea]. Available: https://www.ietf.org/rfc/rfc1543.txt.

[2] K. Scott, S. Burleigh, «RFC 5050 Bundle Protocol Specification,» November 2007. [En línea]. Available: https://tools.ietf.org/html/rfc5050.

[3] United Nations Office for Outer Space Affairs, International Space Law, New York: United Nations, 2017. [En línea]. Available: http://www.unoosa.org/res/oosadoc/data/documents/2017/stspace/stspace61rev_2_0_html/V1703167-SPANISH.pdf

[4] International Telecommunication Union, Measuring the Information Society Report, 2018. [En línea]. Available: https://www.itu.int/dms_pub/itu-d/opb/ind/D-IND-ICTOI-2018-SUM-PDF-S.pdf

[5] A. Schlesinger, B. M. Willman, L. Pitts, S. R. Davidson , W. A. Pohlchuck, «Delay/Disruption Tolerant Networking for the International Space Station (ISS),» de *IEEE Aerospace Conference*, Montana, EEUU, 2017.

[6] K. L. Torgerson, L. Clare, S. Wang and J. Schoolcraft , «The Deep Impact Network Experiment Operations Center,» de *IEEE Aerospace conference*, Montana, EEUU, 2009.

[7] D. M. Boroson, J. J. Scozzafava, D. V. Murphy, B. S. Robinson and M. I. T. Lincoln, «The Lunar Laser Communications Demonstration (LLCD),» de *2009 Third IEEE International Conference on Space Mission Challenges for Information Technology*, California, EEUU, 2009.

[8] F. Warthman, Warthman Associates, «Delay- and Disruption-Tolerant Networks (DTNs): A Tutorial,» 14 Septiembre 2015. [En línea]. Available: http://ipnsig.org/wp-content/uploads/2015/09/DTN_Tutorial_v3.2.pdf.

[9] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, H. Weiss, «RFC 4838 Delay-Tolerant Networking Architecture,» Abril 2007. [En línea]. Available: https://tools.ietf.org/html/rfc4838.

[10] L. Wood, J. McKim, W. Eddy, W. Ivancic, C. Jackson, «IETF Draft. Using Saratoga with a Bundle Agent as a Convergence Layer for Delay-Tolerant Networking,» 6 Octubre 2012. [En línea]. Available: https://tools.ietf.org/html/draft-wood-dtnrg-saratoga-11.

[11] M. Ramadas, S. Burleigh, S. Farrell, «RFC 5326 Licklider Transmission Protocol - Specification,» Septiembre 2008. [En línea]. Available: https://tools.ietf.org/html/rfc5326.

[12] K. Massri, A. Vernata, A. Vitaletti, «Routing protocols for delay tolerant networks: a quantitative evaluation,» de *Proceedings of the 7th ACM workshop on Performance monitoring and measurement of heterogeneous wireless and wired networks*, Paphos, Chipre, 2012.

[13] S. Symington, S. Farrell, H. Weiss, P. Lovell, «RFC 6257 Bundle Security Protocol Specification,» Mayo 2011. [En línea]. Available: https://tools.ietf.org/html/rfc6257.

[14] Lori Keesey, «Disruption Tolerant Networking to Demonstrate Internet in Space,» 16 Julio 2018. [En línea]. Available: https://www.nasa.gov/feature/goddard/2018/disruption-tolerant-networking-to-demonstrate-internet-in-space.

[15] E. Mahoney, «New Solar System Internet Technology Debuts on the International Space Station,» 21 Junio 2016. [En línea]. Available: https://www.nasa.gov/feature/new-solar-system-internet-technology-debuts-on-the-international-space-station.

[16] F. A. Davis, J. K. Marquart, G. Menke, «Benefits of Delay Tolerant Networking for Earth science missions,» *2012 IEEE Aerospace Conference*, Big Sky, Montana, EEUU, 2012, pp. 1-11. [En línea]. Available: https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20120002041.pdf.

[17] S. Parikh, R. C. Durst, «Disruption tolerant networking for Marine Corps CONDOR,» *MILCOM 2005 - 2005 IEEE Military Communications Conference*, Atlantic City, Nueva Jersey, EEUU, 2005, pp. 325-330. [En línea]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1605705&isnumber=33743.

[18] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. S. Peh, D. Rubenstein, «Energy-Efficient Computing For Wildlife Tracking: Design Tradeoffs and Early Experiences with ZebraNet,» de *International Conference on Architectural Support for Programming Languages and Operating Systems*, San Jose, California, EEUU, 2002.

[19] T. Small, Z. J. Haas, «The shared wireless infostation model: a new ad hoc networking paradigm,» de *Mobile and Ad Hoc Networking and Computing*, Annapolis, Maryland, Estados Unidos, 2003.

[20] A. Pereira da Silva, S. Burleigh, K. Obraczka, Delay and Disruption Tolerant Networks: Interplanetary and Earth-Bound, CRC Press, 2018, p. 18.

[21] Joel J. P. C. Rodrigues, Advances in Delay-tolerant Networks (DTNs), Woodhead Publishing, 2014, p. 254.

[22] A. S. Pentland, R. Fletcher, A. Hasson, «DakNet: Rethinking Connectivity In Developing Nations,» *Computer,* vol. 37, nº 1, pp. 78-83, Junio 2004. [En línea]. Available: https://www.ltu.se/cms_fs/1.11736!/file/Summary_N4C_Period_2.pdf

[23] N4C Management, «Networking for Communications Challenged Communities (N4C) Architecture, Test Beds and Innovative Alliances,» 2010.

[24] M. J. Khabbaz, W. F. Fawaz, C. M. Assi, «Probabilistic Bundle Relaying Schemes in Two-Hop Vehicular Delay Tolerant Networks,» *IEEE Communications Letters,* vol. 15, nº 3, pp. 281-283, 2011.

[25] W. Eddy, E. Davies , «RFC 6256 Using Self-Delimiting Numeric Values in Protocols,» Mayo 2011. [En línea]. Available: https://tools.ietf.org/html/rfc6256.

[26] Juan A. Fraire, Pablo G. Madoery, Jorge M. Finochietto, Guillermo Leguizamn, «An Evolutionary Approach Towards Contact Plan Design for Disruption-Tolerant Satellite Networks,» *Applied Soft Computing,* pp. 446-456, 1 Marzo 2017.

[27] L. Yun, C. Xinjian, L. Qilie, Y. Xiaohu, «A Novel Congestion Control Strategy in Delay Tolerant Networks,» de *2010 Second International Conference on Future Networks*, Sanya, Hainan, China, 2010.

[28] Yun Li, Ling Zhao, Zhanjun Liu, Qilie Liu, «N-Drop: congestion control strategy under epidemic routing in DTN,» de *IWCMC '09 Proceedings of the 2009 International Conference on Wireless Communications and Mobile Computing: Connecting the World Wirelessly*, Leipzig, Alemania, 2009.

[29] E. Coe, C. Raghavendra, «Token Based Congestion Control for DTNs,» de *2010 IEEE Aerospace Conference*, Big Sky, Montana, EEUU, 2010.

[30] M. J. Khabbaz, C. M. Assi, W. F. Fawaz, «Disruption-Tolerant Networking: A Comprehensive Survey on Recent Developments and Persisting Challenges,» *IEEE Communications Surveys & Tutorials,* vol. 14, nº 2, pp. 607-640, 2012.

[31] L. Wood, W. M. Eddy, W. Ivancic, J. McKim, C. Jackson, «Saratoga: a Delay-Tolerant Networking convergence layer with efficient link utilization,» de *2007 International Workshop on Satellite and Space Communications*, Salzburgo, Austria, 2007.

[32] H. Kruse, «IRTF Draft. UDP Convergence Layers for the DTN Bundle and LTP Protocols,» 19 Noviembre 2008. [En línea]. Available: https://tools.ietf.org/html/draft-irtf-dtnrg-udp-clayer-00

[33] M. Demmer, J. Ott, «RFC 7242 Delay-Tolerant Networking TCP Convergence-Layer Protocol,» Junio 2014. [En línea]. Available: https://tools.ietf.org/html/rfc7242.

[34] U.S. Naval Research Laboratory, «Common Open Research Emulator (CORE),» [En línea]. Available: https://www.nrl.navy.mil/itd/ncs/products/core.

[35] «Wireshark,» [En línea]. Available: https://www.wireshark.org/.

[36] «IBR-DTN Implementation,» [En línea]. Available: https://github.com/ibrdtn/ibrdtn.

[37] S. Symington, S. Farrell, H. Weiss, P. Lovell, «RFC 6257 Bundle Security Protocol Specification,» Mayo 2011. [En línea]. Available: https://tools.ietf.org/html/rfc6257.

[38] Project Management Institute, Guía de los fundamentos para la dirección de proyectos (PMBOK), Sexta ed., Septiembre 2017.

[39] Johannes Morgenroth, «IBR-DTN - A modular and lightweight implementation of the bundle protocol.,» [En línea]. Available: https://github.com/ibrdtn/ibrdtn.